Opinnäytetyö (AMK)

Tietotekniikan koulutusohjelma

Ohjelmistotuotanto

2012

Joonas Lindholm

WINDOWS PHONE 7 - SOVELLUSKEHITYKSEN PERUSTEET JA YKSINKERTAISEN TWITTER-SOVELLUKSEN KEHITYS

TURUN AMMATTIKORKEAKOULU
TURKU UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖ (AMK) | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Tietotekniikka | Ohjelmistotuotanto

Marraskuu 2012 | 59

Ohjaaja: Tiina Ferm

Joonas Lindholm

WINDOWS PHONE 7 -SOVELLUSKEHITYKSEN PERUSTEET JA YKSINKERTAISEN TWITTER-SOVELLUKSEN KEHITYS

Tässä opinnäytetyössä kuvataan Windows Phone 7 -mobiilikäyttöjärjestelmää. Työssä käsitellään käyttöjärjestelmän yleistiedot ja tekniikat, joita sovelluskehittäjät voivat hyödyntää. Tämän lisäksi esitellään kehitykseen läheisesti liittyvät kehitystyökalut sekä tarkastellaan niiden käyttöä rikkaan sovelluksen luomisessa.

Windows Phone 7 on nopeasti kasvava ja kehittyvä Microsoftin luoma mobiilikäyttöjärjestelmä. Käyttöjärjestelmä on julkaistu vuonna 2010, jolloin myös sovelluskehittäjät pääsivät rakentamaan alustan päälle omia sovelluksiaan. Sovelluskehityksessä käytettävät tekniikat jaetaan kahteen osaan. Ensimmäinen osista on Silverlight-tekniikka jolla luodaan rikkaita interaktiivisia sovelluksia, ja toinen pelikehitykseen tarkoitettu XNA Framework.

Opinnäytetyössä tarkastellaan erityisesti Microsoftin mobiilisovelluskauppaa, jonka välityksellä sovelluskehittäjät voivat jakaa omia tuotoksiaan kuluttajille. Julkaisuprosessin sisältämä tarkistus sertifioi sovelluksen sekä tarkistaa sen laadun ja tietoturvallisuuden.

Opinnäytetyön empiriaosassa luodaan opittujen asioiden pohjalta oma mobiilisovellus. Sovellus yhdistetään Twitter mikroblogipalvelun tarjoamaan rajapintaan, jonka avulla sovellus asetetaan näyttämään palvelussa esillä olevia tietoja.

Puhelinsovellus antaa käyttäjälle mahdollisuuden tarkastella kenen tahansa Twitter-palveluun rekisteröityneen käyttäjän julkaisemia viestejä. Sovellusta voidaan käyttää kaikkialla, missä puhelin on mahdollista yhdistää internetiin.

ASIASANAT:

Windows Phone 7, Visual Studio, sovelluskehitys, Microsoft, mobiili, Silverlight, XNA

BACHELOR´S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Information Technology | Software Engineering

November 2012 | 59

Instructor Tiina Ferm

Joonas Lindholm

WINDOWS PHONE 7 SOFTWARE DEVELOPMENT BASICS AND CREATING A SIMPLE TWITTER APPLICATION

The aim of this thesis is to present the Windows Phone 7 mobile operating system. It describes the general information and technologies of the operating system that are made available for the developers. Moreover, it also covers the tools that are closely related in creating rich mobile applications.

Windows Phone 7 is a rapidly growing mobile operating system created by Microsoft. The platform was published in 2010, at which time the developers were also given the required tools for application development. Technologies used in application development are divided into two categories. First there is Silverlight -technology, which allows developers to create rich, interactive applications, and the second one is XNA Framework for mobile game development.

The paper reviews in particular the application marketplace that is provided for developers who wish to publish their applications. Publishing process certificates the application and also evaluates its quality and security issues.

The second part of this thesis concentrates on creating an example application to Windows Phone platform. The application is connected to Twitter social networking services via Twitter API. After connecting to the API the application is updated to show information from the service.

The application provides its user a possibility to view status updates posted on Twitter by any registered user. The application is usable anywhere the mobile phone can connect to the internet.

KEYWORDS:

Windows Phone 7, Visual Studio, software development, Microsoft, mobile, Silverlight, XNA

SISÄLTÖ

KUVAT

TAULUKOT

KÄYTETYT LYHENTEET

WP7	lyhenne jota käytetään yleisesti Windows Phone 7 -käyttöjärjestelmästä
XAML	käyttöliittymien kuvaamiseen käytetty kieli Windows Phonessa
.NET	microsoftin kehittämä ohjelmistokomponenttikirjasto
C#	microsoftin luoma ohjelmointikieli
IDE	ohjelmointiympäristö, jolla toteutetaan ohjelmistoja
SDK	ohjelmiston suunnitteluun vaadittavat kehitystyökalut sisältävä paketti
API	ohjelmointirajapinta, jonka avulla ohjelmat voivat tehdä pyyntöjä ja vaihtaa tietoja keskenään
GPS	satelliittien välityksellä paikkatietoja antava paikannusjärjestelmä
OData	protokolla, jonka avulla voidaan jakaa tietoa eri laitteille
OAuth	rajapintoihin tunnistautumisee suunniteltu protokolla

JOHDANTO

Microsoft julkaisi kehittämänsä Windows Phone 7 -mobiilikäyttöjärjestelmän 2010 Mobile World Congress -nimisessä mobiilialan suurtapahtumassa. Tämä mobiilikäyttöjärjestelmä luotiin uudistamaan täysin Microsoftin entinen mobiilikäyttöjärjestelmä. Sen tarkoituksena on olla vastaus älypuhelinten käyttöjärjestelmien nopeaan kehitykseen ja kilpailuun.

Tämän työn tarkoituksena on tehdä lukijalle tutuksi, millainen on Windows Phone 7 -käyttöjärjestelmä, sekä millaisia työkaluja Microsoft tarjoaa sovelluskehittäjille. Työn alussa esitellään yleisesti käyttöjärjestelmän nykytilanne, jotta lukija saa kuvan siitä, millaisessa asemassa Windows Phone 7 on mobiilimarkkinoilla. Lukijalle esitellään myös uusimman käyttöjärjestelmäversion tuomia tärkeitä ominaisuuksia, joita käytämällä voidaan entisestään parantaa alustalle laadittuja sovelluksia.

Työssä suunnitellaan ja toteutetaan mobiilisovellus esiteltyjä tekniikoita käyttäen. Sovelluksen käyttöliittymä ja ulkoasu pyritään luomaan yksinkertaiseksi, mutta kuitenkin XAML-kieltä käyttäen. Sovellus luodaan työssä esiteltävää Microsoft Silverlight -tekniikkaa käyttäen. Lopuksi sovelluksen käyttökokemusta parannetaan animaatioiden avulla. Animaatiot toteutetaan Expression Blend 4 -työkalulla.

1 WINDOWS PHONE 7

1.1 Windows Phone 7 julkaisu ja kehitys

Windows Phone 7 on Microsoftin vuoden 2010 helmikuussa julkaisema mobiilikäyttöjärjestelmä. Se on kehitetty seuraajaksi Windows Mobile käyttöjärjestelmälle. Windows Phone 7:n kehittäminen aloitettiin jo vuonna 2004 koodinimellä ”Photon”. Projekti kuitenkin keskeytettiin vuonna 2008. Myöhemmin Microsoft teki päätöksen täysin uudenlaisen mobiilikäyttöjärjestelmän kehittämisen aloittamisesta. Organisaatiomuutosten jälkeen WP7:n kehitys aloitettiin täysin puhtaalta pöydältä, vaikkakin uuden käyttöjärjestelmän ytimessä on tästä huolimatta käytössä rippeitä Windows CE:n hyväksi todetuista koodeista. Ohjelmistokehityksen näkökulmasta katsottuna suurin muutos on se, että vanhat Windows Mobile -sovellukset eivät ole tuettuina uudessa Windows Phonessa, vaan uusien sovellusten ohjelmointi tapahtuu käyttäen C#- tai Visual Basic -kieliä sekä Silverlight ja XNA-tekniikoita. Windows Phonesta on tällä hetkellä olemassa kaksi pääversiota, Windows Phone 7.0 sekä Mango -koodinimellä kulkeva Windows Phone 7.5. Microsoftin mobiilikäyttöjärjestelmää laitteissaan käyttävät tällä hetkellä ainakin seuraavat puhelinvalmistajat:

- Nokia
- HTC
- LG
- Samsung
- DELL
- Toshiba
- ZTE
- Acer
- Fujitsu

Näiden valmistajien lisäksi myös ASUS on ilmoittanut kiinnostuksensa WP7:aa kohtaan, ja siltä odotetaankin soveltuvaa puhelinmallia älypuhelinmarkkinoille. [1, s. 12] [2, 3, 4]

Vuoden 2012 toisella neljänneksellä Windows Phone 7 -älypuhelinten myynti oli Gartnerin mukaan noin 4 miljoonaa myytyä puhelinta (kuva 1). Jos tätä tietoa verrataan 2012 ensimmäiseen neljännekseen voidaan havaita, että Windows Phone -älypuhelinten myynti on kasvanut vuoden 2012 ensimmäisen ja toisen neljänneksen välillä noin puolitoista kertaiseksi. Windows Phone on tällä hetkellä siis hyvin kasvava mobiilikäyttöjärjestelmä, joten sovelluskehittäjien ei kannata unohtaa käyttöjärjestelmän tuomia mahdollisuuksia. [5]

Gartner: World-Wide Smartphone Sales (Thousands of Units)

Quarter	Windows Mobile[6]	RIM	Symbian[7]	iOS	Android[8]	Bada	Windows Phone[9]	Other	Total Smartphones	Total Phones
2012 Q2 [10]		7,991	9,072	28,935	98,529	4,209	4,087	863	153,686	419,008
2012 Q1 [11]		9,939	12,467	33,121	81,067	3,842	2,713	1,243	144,392	419,108

Kuva 1. Älypuhelinten myynti 2012. [5]

1.2 Laitteistovaatimukset

Windows Phonen laitteistovaatimukset päivittyivät sittemmin hyvin paljon entistä joustavammiksi. Tällä hetkellä laitteistovaatimukset molemmille julkaistuille Windows Phone -versioille ovat samat. Microsoft päivittää viimeisimmät laitteistovaatimukset MSDN-kehittäjäpalveluunsa osoitteeseen www.msdn.com. Taulukossa 1 on kuvattu Microsoftin 16.8.2012 asettamat minimivaatimukset.

Taulukko 1. Windows Phone 7 laitteiston minimivaatimukset.

Ominaisuus	Minimivaatimus
Muisti	vähintään 256MB keskusmuisti / 8GB flash-muisti.
Näytönohjain (GPU)	DirectX 9 -laitteistokiihdytystä tukeva ohjain.
Verkkoyhteydet	GSM/CDMA, GPRS, 3G, WLAN.
Näyttö	Kapasitiivinen kosketusnäyttö jossa tuki 4 pisteen kosketukselle sekä 800x480 resoluutio.
Sensorit	Kiihtyvyysanturi.
Muut vaatimukset	Takaisin, Aloita, Etsi -painikkeet.

Kuten taulukosta 1 voidaan havaita, ovat tämän hetkiset valmistajille asetetut minimivaatimukset hyvin alhaiset. Lisäksi joustavuus laitevaatimuksissa antaa myös puhelinvalmistajille mahdollisuuden omiin lisäyksiin, joista yleisimpänä

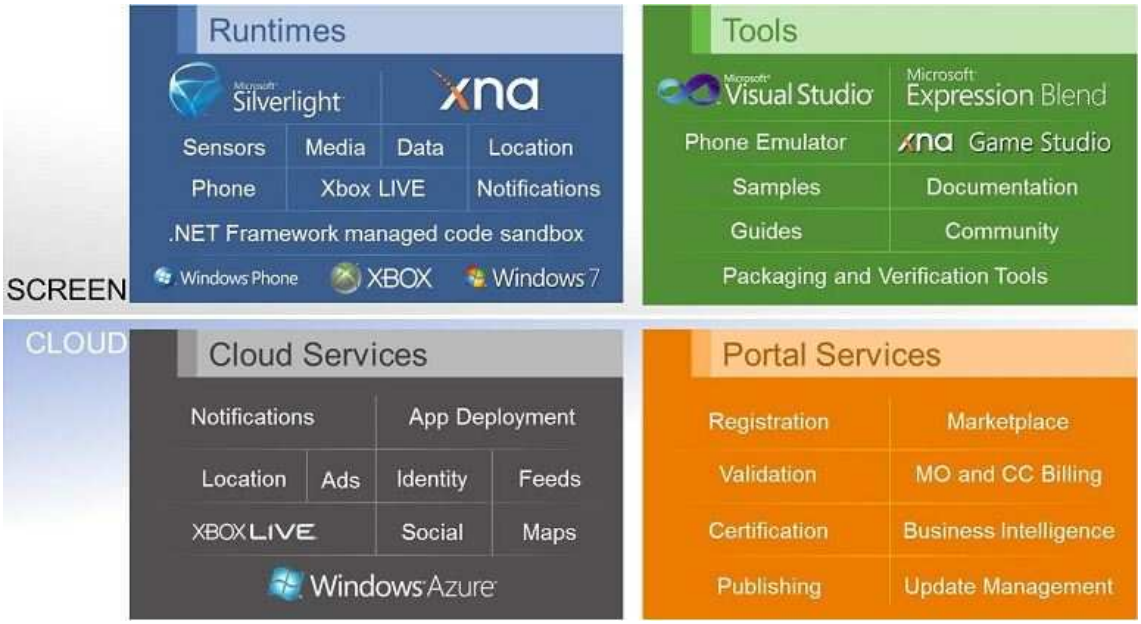
mainittakoon esimerkiksi Bluetooth-yhteyden lisääminen. Microsoftin alhaisten laitevaatimusten toisena tarkoituksena on edistää osaltaan puhelinvalmistajien kynnystä valmistaa Windows Phone -käyttöjärjestelmälle soveltuvia puhelimia. Toki on itsestäänselvyys, että laitteistovaatimukset nousevat ajan myötä käyttöjärjestelmän kehittyessä.

Suomessa tällä hetkellä myytävien Windows Phone -puhelinten hintaskaala vaihtelee noin 180 euron ja 500 euron välillä, joten hinnat ovat samalla tasolla kilpailijoiden puhelinmallien kanssa. [1, 4, 6]

1.3 Alustan arkkitehtuurin pääkomponentit

Windows Phone -alustan arkkitehtuuri (kuva 2.) koostuu neljästä eri pääkomponentista, joita ovat

1. ajonaikaiset (Runtimes)
2. työkalut (Tools)
3. pilvipalvelut (Cloud Services)
4. portaalipalvelut (Portal Services)



Kuva 2. Windows Phone -alustan neljä pääkomponenttia. [7]

Runtimes

Runtimes komponentti kattaa Silverlight- ja XNA -frameworkit, sekä Windows Phone -erityispiirteet yhdistettynä ympäristöön, jonka päälle voidaan rakentaa turvallisia ja graafisesti rikkaita sovelluksia. Käytännössä siis näiden komponenttien avulla ohjelmoija voi tuottaa haluamiaan ohjelmistoja puhelimelle. Tämän työn myöhemmässä vaiheessa tutustutaan tarkemmin Silverlight sekä XNA -komponenttikirjastoihin ja niiden käyttömahdollisuuksiin.

Ensimmäinen pääkomponentti sisältää myös erilaisia sensoreita kuten multitouch sensorin, kiihtyvyysanturin, kompassin, gyroskoopin ja mikrofonin, joiden palauttamaa dataa ohjelmoija voi käyttää API:en (ohjelmointirajapinta) avulla. [7]

Komponentti tarjoaa myös kehittäjälle eristetyn tallennuksen, joka antaa kirjoitetuille ohjelmille paikan jonne tallentaa, ja josta hakea tietoa. Tämä on toteutettu ns. hiekkalaatikkotyylisesti, jolloin jokaiselle ohjelmistolle on tarjolla oma hiekkalaatikkonsa. Tällä estetään sovelluksen pääsy taustalla toimivan käyttöjärjestelmän tiedostoihin sekä tietojen väärinkäyttö ja tuhoutuminen. [7]

Tehdyille ohjelmistoille tarjotaan myös mahdollisuus käyttää hyväkseen puhelimen antamia sijaintitietoja. Näitä ovat tieto nykyisestä sijainnista, sijainnin muutoksesta, suunta- ja nopeustieto sekä etäisyys paikkojen välillä. [7]

Työkalut

Työkalut, jotka Microsoft tarjoaa Windows Phone kehittäjille, esitellään tämän opinnäytetyön toisessa luvussa.

Pilvipalvelut

Pilvipalvelu käsitteenä tarkoittaa ”pilvessä” olevia, eli yksinkertaistettuna internetissä olevia ohjelmapalveluita. Pilvipalvelujen käyttö mobiililaitteissa auttaa

parantamaan niiden suorituskykyä ja akun kestoa, koska suurin osa tiedon käsittelystä tehdään itse pilvipalvelussa. Windows Phone -sovellusalusta tarjoaa kehittäjälle käytettäväkseen seuraavanlaiset pilvipalvelurajapinnat:

- ilmoitukset (Notifications)
- sijainti pilvipalvelu (Location Cloud Services)
- sosiaaliset mediat, Syötteet sekä Kartta -palvelut (Identity, Feeds, Social, and Maps Services)
- mainonta (Microsoft Advertising for Windows Phone) [7].

Ilmoitukset tarjoavat pilvessä olevan ilmoituspalvelun, jonka kautta voidaan tarvittaessa lähettää haluttuja ilmoituksia sovelluksille. Pilvestä tulevilla ilmoituksilla eliminoidaan puhelimen jatkuva ja runsaasti akkua kuluttava tilan päivitys. [7]

Sijaintipalvelu toimii yhdessä puhelimen paikkatieto API:n kanssa, jolloin puhelimen loppukäyttäjän ei tarvitse välittää sen toiminnasta. Palvelu käyttää toimiakseen puhelimen käytössä olevia verkkoyhteysmahdollisuuksia sekä sisäänrakennettua GPS:ää. [7]

Suuri määrä erilaisia verkkopalveluita on yhteydessä pilveen ja näiden avulla kuluttajat voivat olla vuorovaikutuksessa sosiaalisen median kautta, vastaanottaa haluamiaan syötteitä ja käyttää tarvittaessa karttoja navigaatioon. Niiden avulla kehittäjät voivat halutessaan rikastuttaa omien ohjelmistojensa käyttäjäkokemusta. [7]

Windows Phone tarjoaa myös kehittäjälleen mahdollisuuden näyttää mainoksia kirjoittamissaan sovelluksissa. Tätä varten Microsoft on luonut SDK:n, jonka avulla mainosbannerin lisääminen ohjelmaan onnistuu vaivattomasti. [7]

Portaalipalvelut

Portaalipalvelu pitää käytännössä sisällään Windows Phone Marketplace -sovelluskaupan. Sovelluskaupan kautta kehittäjät voivat jakaa luomaansa sovellusta kaikille Windows Phone -puhelimia käyttäville kuluttajille. Prosessin yhteydessä julkaistu sovellus myös tarkistetaan ja sertifioidaan ennen jakamista. Windows Phone Marketplacen toimintaa esitellään tarkemmin opinnäytetyön luvussa kolme. [7]

1.4 Käyttöjärjestelmän yleiset ominaisuudet

Luodessaan Windows Phonen koodinimellä ”Metro” kulkevaa käyttöliittymää ja sen ulkoasua, on Microsoft panostanut hyvin paljon yksinkertaiseen käytettävyyteen ja selkeyteen. Tämän suunnittelufilosofian myötä on syntynyt käyttöliittymän näkyvin piirre, aloitusnäkymä eli Start Screen. Aloitusnäkymä koostuu niin sanotuista tiilistä (tiles). Tiilten pääasiallinen tarkoitus on toimia pikakuvakkeina, eli niiden avulla voidaan käynnistää haluttu sovellus. Tiilet ovat lisäksi interaktiivisia, eli ne pystyvät näyttämään tilannetietoja esimerkiksi käyttäjän saadessa uuden tekstiviestin. Tiiliä on kahta eri kokoa, joista toinen ulottuu aloitusruudun laidasta laitaan ja toinen vain puoleen väliin asti. Käyttäjän on mahdollista lisätä aloitusruutuun myös valitsemiaan tiiliä. Windows Phone 7:n aloitusruutu on esitelty kuvassa 3.

Kuva 3. Windows Phone -aloitusnäkymä. [3]

Tiilien kautta käyttäjä pääsee myös käsiksi käyttöliittymän keskeisimpiin osiin, jotka on kerätty isommiksi kokonaisuuksiksi. Näitä kokonaisuuksia kutsutaan nimellä keskukset (Hubs). Windows Phonessa näitä keskuksia on kuusi kappaletta:

- ihmiset
- kuvat
- pelit
- musiikki ja videot
- sovelluskauppa
- office [8].

Aukaistessaan jonkin keskuksen käyttäjä pääsee tarkastelemaan laajemmin keskuksen sisälle koottuja kokonaisuuksia. Esimerkiksi avatessaan Kuvat -keskuksen voi käyttäjä katsella ja hallinnoida kaikkia puhelimessaan olevia multimediatiedostoja kuvista videoihin (kuva 4). Samalla käyttäjä pääsee myös käsiksi vaikkapa PC:llä oleviin kuvatiedostoihinsa SkyDrive -palvelun avulla, sekä

näkee ystäviensä uusimmat kuvapäivitykset sosiaalisista medioista. Lisäksi käyttäjä pääsee Kuvat -keskuksen avulla käynnistämään sovelluskaupasta ladattuja sovelluksia, jotka liittyvät jollain tavalla kyseessä olevan keskuksen toimintaan. Esimerkkinä tästä on Imgur Photo Uploader -sovellus, jolla käyttäjä voi ladata minkä tahansa kuvatiedoston puhelimestaan http://imgur.com -sivustoon.

Kuva 4. Pictures Hub eli Kuvat -keskus. [9]

1.5 Uusimmat ominaisuudet sovelluskehittäjille

Windows Phonen version 7.5 myötä sovelluskehittäjät saivat pitkän listan merkittäviä uudistuksia. Tämän päivityksen myötä myös kehitystyökalupaketti Windows Phone SDK kasvatti versiointiaan numeroon 7.1.

Ylivoimaisesti merkittävin ominaisuus sovelluskehittäjille on uusi parannettu moniajo. Ennen Mango -päivitystä käyttöjärjestelmä sammutti automaattisesti sovellukset, joista käyttäjä poistui paluunäppäimellä. Mangossa kuitenkin käyttöjärjestelmä antaa mahdollisuuden yhden sovelluksen tausta-ajoon ns. lepotilassa. Lepotilassa olevat sovellukset eivät kuitenkaan voi olla tausta-ajossa ikuisuuksia, vaan tausta-ajo mahdollisuuden kesto riippuu laitteen muistin mää-

rästä. Useampia sovelluksia ei ole mahdollista ajaa taustalla samanaikaisesti virrankulutuksen optimoimiseksi. [10]

Toinen sovelluskehittäjiä varmasti kiinnostava uutuus oli Visual Basic tuen lisääminen aiemmin ainoana vaihtoehtona toimineen C# -kielen ohelle. [11]

Mangon myötä Windows Phonen tietoliikenneyhteyksiä vahvistettiin TCP- ja UDP-protokollien kanssa. Tämän päivityksen myötä sovellukset voivat hoitaa tietoliikennettään kyseisten protokollien avulla luoden sovelluskehittäjälle mahdollisuuden vaikkapa moninpelien luomiseen. [11]

Sovelluskehittäjien saataville päivityksessä tulivat myös tarkemmat verkko- ja laiteinformaatiot. Verkkoinformaatiohin kuului operaattorin nimitieto, verkkoyhteyden tyyppi ja datayhteyden tila. Laiteinformaatiossa sen sijaan saataville tuli puhelimen merkki ja malli, muistinkäyttö ja virrankäyttötila. [11]

Push-ilmoitukset (push notifications) saivat oman osansa parannuksista. Käytännössä parannukset antavat sovellukselle ja sen käyttäjälle mahdollisuuden saada ilmoituksia tapahtuvista muutoksista, jos sovelluskehittäjä niin haluaa. Push -ilmoituksilla voidaan myös päivittää interaktiivisten tiilten tilaa, tai antaa sovelluksen käyttäjälle ilmoitus käyttöliittymän yläreunaan sille varattuun kohtaan. [1, s.59] [11]

Interaktiiviset tiilet päivittyivät kaksipuolisiksi. Mangossa tiilet voivat kääntyä aloitusruudulla ja antaa kääntöpuolella tarkempaa informaatiota, esim. esikatselun saapuneesta viestistä sekä luoda samalla visuaalista iloa käyttäjälle. [1, s.59] [11]

Mangossa Silverlightin ja XNA Frameworkin yhteiskäyttöä on pyritty parantamaan antamalla mahdollisuus sekoittaa tekniikoita keskenään. Esimerkiksi XNA-sovellukseen voidaan Mangon myötä lisätä Silverlight -tekniikan avulla tehtyjä valikkoja sekä päinvastoin. [11]

Paikallinen tietokanta on myös yksi Mangon tuomista uudistuksista. Se on kevyeen käyttöön soveltuva SQL-tietokanta, jonka käyttö onnistuu LINQ-lauseilla. [1, s. 63]

Myös aiemmin työssä mainittu mainontapalvelu tuli Windows Phoneen vasta Mango -versiossa. Mainosten lisääminen omiin sovelluksiin antaa käyttäjälle mahdollisuuden ansaita rahaa sovellukseensa liitettävillä mainosbannereilla, vaikka sovellus itsessään olisi tarjolla ilmaiseksi. [11]

Ohjeistus sovellusten kokeiluversioiden luontiin on tuotu Mangossa tarkemmin esille. Tällä tavoin kehittäjiä yritetään saada tuottamaan sovelluksistaan kokeiluversioita yhä enemmän. Varsinaisesti kokeiluversioiden luonti jää sovelluskehittäjän harteille siinä mielessä, että hänen on itse päätettävä, kuinka kokeiluversion toteuttaa, ja tämän jälkeen julkaistava kokeiluversio sovelluskaupassa. [1, s. 65]

OData client eli asiakasohjelma tuotiin myös Windows Phoneen Mango -päivityksessä. OData -asiakasohjelmaa käytetään SQL-pohjaisen tietokantadatan kuljettamiseen verkon ylitse. OData luo sovelluksille myös mahdollisuuden lisätä, muokata ja poistaa jonkin SQL-pohjaisen tietokannan tietoja. [1, s. 66]

1.6 Käyttöjärjestelmään liittyvät tekniikat

Yksi käyttöjärjestelmään läheisesti liittyvä tekniikka on Silverlight, joka perustuu .NET 3.0:n mukana tulleeseen Windows Presentation Foundation -tekniikkaan. Se on tarkoitettu visuaalisesti rikkaiden, interaktiivisten käyttöliittymien luomiseen ja niiden julkaisemiseen webissä, työpöytäsovelluksissa ja puhelimissa. Nykyisin Silverlight -tekniikkaa käytetään suurimmaksi osin web-sovellusten tekemiseen. Tämä johtuu siitä, että sen avulla voi erinomaisesti yhdistellä vektorigrafiikkaa, animaatioita, kuvaa, videoita ja ääntä. Tärkeimmät asiat, jotka puolsivat Silverlightin valintaa Windows Phone 7 -käyttöjärjestelmän käyttöliittymätekniikaksi olivat ajonaikaisen ympäristön pieni koko ja alustariippumattomuus. [1, s. 96] [12]

1.6.1 Silverlight for Windows Phone

WP7 älypuhelimissa on käytössä Silverlight for Windows Phone -tekniikka. Se ei kuitenkaan sisällä kaikkia samoja ominaisuuksia Silverlight 4:n kanssa. Näistä suurin osa on karsittu pois pelkästään siksi, että niillä ei ole minkäänlaista käyttöä mobiilikäyttöjärjestelmässä. Silverlight -tekniikkaa käytetään Windows Phone -ohjelmoinnissa käytännössä aina, koska uuden Mango -päivityksen myötä sitä käytetään yleisesti myös XNA-peliohjelmoinnissa. [13]

1.6.2 XNA Framework

XNA Framework on Microsoftin luoma peliohjelmointia helpottava ohjelmistokehys, joka sisältää kokoelman erilaisia luokkakirjastoja ja palveluja. Tämän tekniikan avulla pystytään nopeasti kehittämään pelejä useampaan järjestelmään, joita ovat mm. Windows Phone, XBOX 360 ja Windows 7. XNA:n tärkeimpänä ominaisuutena voidaan pitää ns. pelisilmukkaa, jonka avulla voidaan automatisoida pelin vaatimat päivitys- ja piirtorutiinit. Käytettäessä alustana Windows Phone 7:aa, voidaan ohjelmointikielinä käyttää Visual Basic tai C# -kieltä, mutta ainoastaan näistä jälkimmäinen on virallisesti tuettu. Windows tarjoaa pelikehittäjille työkalun nimeltä XNA Game Studio, jonka uusin versio on 5.0. Windows Phone -pelikehittäjien on kuitenkin mahdollista käyttää XNA:n tarjoamia ominaisuuksia suoraan Visual Studio työkalun kautta. [14,15]

2 KEHITYSTYÖKALUT

2.1 Windows Phone SDK 7.1

Ennen kuin sovelluskehittäjä pääsee työstämään omia sovelluksiaan Windows Phone -käyttöjärjestelmälle, on hänen hankittava itselleen oikeanlaiset kehitystyökalut. Microsoft on helpottanut sovelluskehittäjien tehtävää ja koonnut yhteen isoon pakettiin kaikki Windows Phone -kehitykseen tarvittavat työkalut. Uusin Windows Phone SDK on ladattavissa täysin maksutta Microsoftin Download Centeristä osoitteessa http://www.microsoft.com/en-us/download/details.aspx?id=27570. Kehitystyökalupaketti sisältää seuraavat maksuttomat ohjelmistot:

- Microsoft Visual Studio 2012 Express for Windows Phone
- Windows Phone Emulator
- Windows Phone SDK 7.1 Assemblies
- Silverlight 4 SDK and DRT
- Windows Phone SDK 7.1 Extensions for XNA Game Studio 4.0
- Microsoft Expression Blend SDK for Windows Phone 7
- Microsoft Expression Blend SDK for Windows Phone OS 7.1
- WCF Data Services Client for Windows Phone
- Microsoft Advertising SDK for Windows Phone

Asennusohjelma on ns. web-installer, eli se lataa kaikki yllä listatut ohjelmistot internetistä. Tästä syystä asentaminen saattaa kestää jos käytössä on hidas internet-yhteys. Asennus käynnistyy kuvan 5 mukaisesti painamalla Install Now -näppäintä.

Kuva 5. Windows Phone SDK -asennusohjelma.

Työkalujen asentamisen jälkeen on syytä varmistaa, onko Windows Phone SDK:hon tarjolla uusia päivityksiä. Päivitykset on listattuna yllä mainitussa osoitteessa Download Centerissä. Kaikki päivitykset on hyvä asentaa ennen työkalujen käytön aloittamista.

2.2 Visual Studio 2012 Express for Windows Phone

Visual Studio 2012 Express for Windows Phone on Microsoftin tarjoama pääsovelluskehitysväline WP7 käyttöjärjestelmälle. Se on ns. RAD-ajatukseen perustuva sovelluskehitin, jolla on neljä pääasiallista osaa. Näitä ovat koodieditori, visuaalisten käyttöliittymien suunnitteluosa, kääntäjä ja virheenjäljitystoiminto. Visual Studio -sovelluskehitin tukee useaa ohjelmointikieltä, mutta Windows

Phone -ohjelmoinnissa käytettävissä on ainoastaan C# ja Visual Basic kielet. Uusien sovelluskehittäjien, joilla ei ole aiempaa kokemusta juuri Visual Basic -kielestä, suositellaan aloittamaan suositummalla ja uudemmalla C# -ohjelmointikieltä.

Ennen kehittämisen aloittamista on syytä huomioida, että SDK:n mukana tulevalla Visual Studion Express versiolla on ainoastaan mahdollista kehittää puhelimessa toimivia sovelluksia. Tämä tarkoittaa sitä, että esimerkiksi web-pohjaiset taustajärjestelmät tai vaikkapa sovellusten liittäminen Azure-pilvipalveluun ei ole mahdollista.

Uuden projektin luominen Visual Studiossa

Visual Studion käynnistämisen jälkeen luodaan ensin uusi projekti painamalla aloitus-sivulta New Project -painiketta. Tämän jälkeen valitaan haluttu listalta haluttu ohjelmointikieli, projektimalli ja määritetään projektin nimi (kuva 6). Tässä vaiheessa kehittäjän on mahdollista päättää, haluaako hän tehdä sovelluksensa käyttäen Silverlight -tekniikkaa vaiko vaihtoehtoisesti XNA Game Studio 4.0 -malleja.

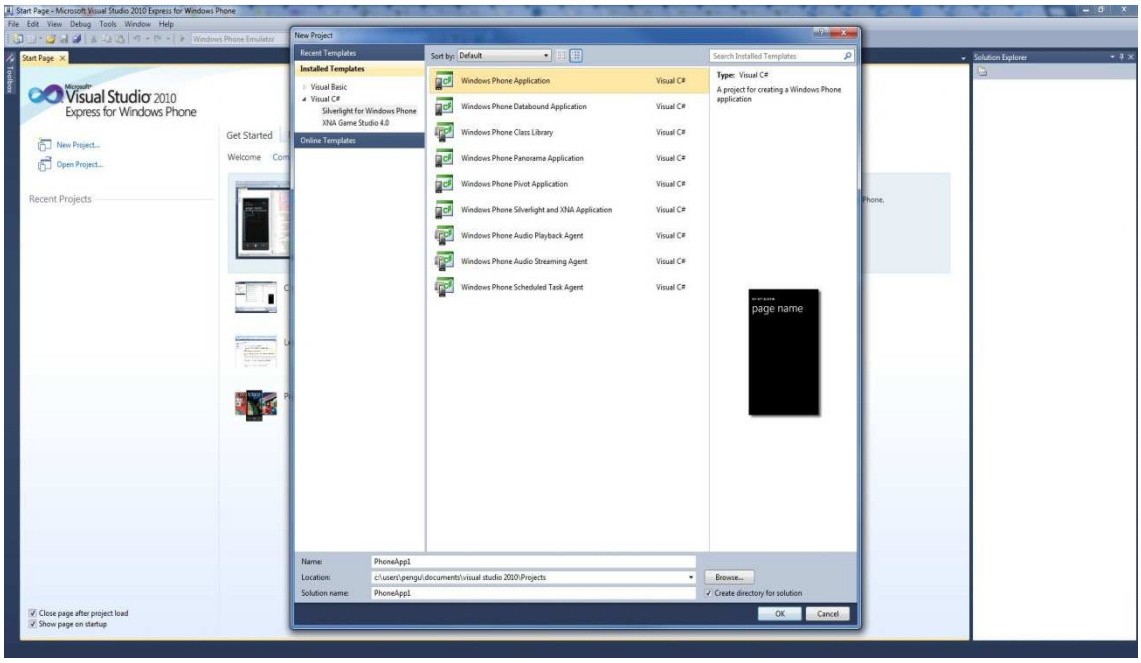
Yleisimmät Silverlight -projektimallit ovat:

- Windows Phone Application – tavallinen yhden sivun sovellus
- Windows Phone Panorama Application – panoraama -kontrollia käyttävä sovellus useammalla sivulla
- Windows Phone Pivot Application – pivot -kontrollia käyttävä sovellus useammalla sivulla

Yleisimmät XNA Game Studio -projektimallit ovat:

- Windows Phone Game (4.0)
- Windows Phone Silverlight and XNA Application

XNA -malleista jälkimmäinen on Mango -versiossa uudistuksena tullut yhdistetty malli, jossa on mahdollista käyttää molempia tekniikoita samanaikaisesti.



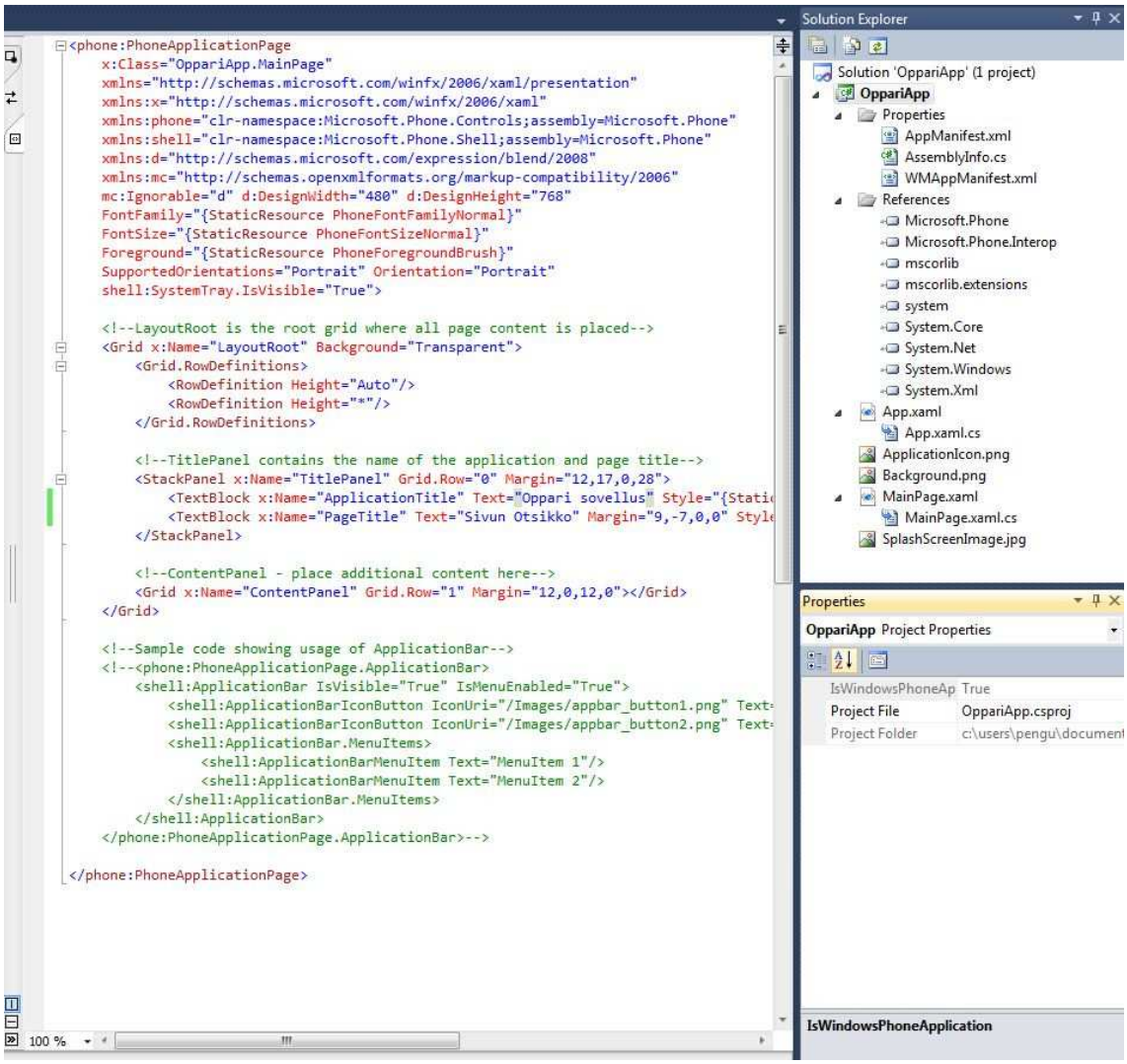
Kuva 6. Uuden projektin luonti Visual Studiossa.

Kun käyttäjä on valinnut haluamansa projektimallin, kysyy Visual Studio mille alustalle sovellus halutaan tehdä. Vaihtoehtoina ovat OS 7.1 sekä OS 7.5 eli Mango. Seuraavaksi käytäjälle avautuu näkymä varsinaisesta kehitysympäristöstä.

Kehitysympäristö koostuu kolmesta osiosta. Ensimmäisessä osassa näkyy visuaalinen kuva sovelluksen etusivusta (kuva 7). Tämä kuva näyttää kehittäjälle visuaaliset muutokset, joita sovellukseen on tehty. Tämä osio toimii myös esikatseluna sille, miltä sovellus näyttää kun käyttäjä avaa sen ensi kertaa Windows Phone -älypuhelimessaan.

Kuva 7. Esikatselukuva sovelluksen sivusta.

Toisessa osassa näkyy XAML-koodi, jolla juuri visuaalista osaa ja sen asetteluja voidaan muokata (kuva 8). XAML-kieli on XML -pohjainen kieli, jota käytetään Windows Phone -ohjelmoinnissa nimenomaan käyttöliittymän suunnitteluun. XAML-kielessä, kuten XML-kielessä, kaikki käyttöliittymän kontrollit kuvataan elementeillä ja niiden attribuuteilla. Nämä taas vastaavat varsinaisessa ohjelmakoodissa luotuja olioita ja niille annettuja ominaisuuksia. [1, s. 97]

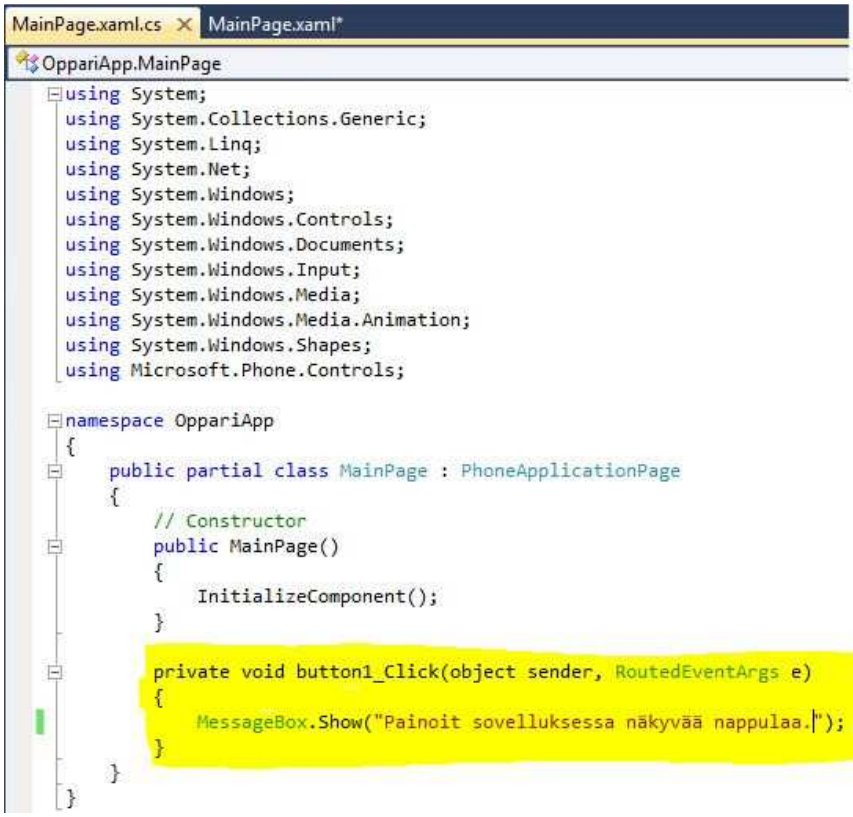


Kuva 8. XAML-koodieditori ja Solution Explorer Visual Studiossa.

Kolmas näkyvä osa on ns. Solution Explorer (kuva 8). Solution Explorerin avulla käyttäjä voi hallita kaikkia projektiinsa kuuluvia tiedostoja ja tarpeen vaatiessa lisätä niitä. Halutessa käyttäjä saa esiin Properties valikon, joka sijoittuu Solution Explorerin alareunaan (kuva 8). Properties valikko voidaan avata F4 -näppäinkomennolla.

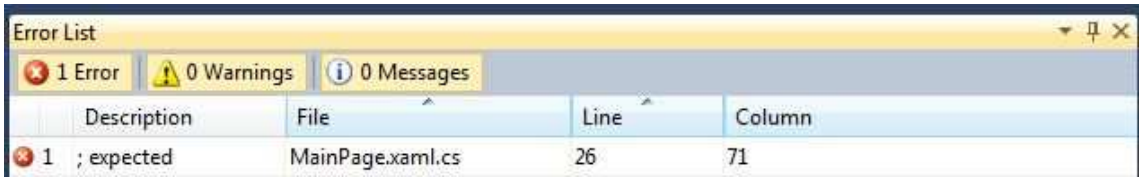
Windows Phone -kontrollit ovat oletuksena piilotettu. Käyttäjä saa ne näkyviin viemällä hiiren sovelluksen vasempaan reunaan kohtaan, jossa lukee Toolbox. Kontrolleja voidaan lisätä sovelluksiin raahaamalla ne kontrolli-valikosta sovelluksen sivun esikatseluun. Kun haluttu kontrolli on raahattu ja pudotettu sivulle, voidaan sille kirjoittaa haluttua C#-koodia kaksoisklikkaamalla sitä. Tämä aukai-

see piilossa olleen C#-koodieditorin. Koodi kirjoitetaan editorissa halutun kontrollin kohdalle. Oletuksena uudet lisätyt kontrollit listautuvat koodissa alimmaksi (kuva 9).



Kuva 9. C#-editori, jossa keltaisella lisätyn kontrollin koodi.

Kun haluttu koodi on kirjoitettu, voidaan sovellus rakentaa ja ajaa Windows Phone -emulaattorissa. Sovelluksen rakentaminen onnistuu painamalla näppäintä F6. Rakentaessa sovellusta kääntäjä tarkistaa ja etsii koodista löytyviä virheitä. Jos koodi sisältää virheitä, ilmoittaa Visual Studio virheet virhelistassa koodieditorin alla. Virheenjäljitystoiminto esitelty kuvassa 10.



Kuva 10. Virheenjäljitystoiminto Visual Studiossa.

Virheenjäljitystoiminto kertoo ohjelmoijalle koodista löytyvien virheiden tiedot. Näitä tietoja ovat virheen kuvaus, tiedosto jossa virhe on, rivinumero ja sarakkeen numero.

Kun mahdolliset virheet on korjattu ja sovellus on rakennettu onnistuneesti, on se mahdollista emulaattorissa testikäyttöä varten. Pudotusvalikko Visual Studion päävalikon alla kertoo mihin koodi ajetaan. Vaihtoehtoina ovat Windows Phone -emulaattori sekä Windows Phone -laite. Kun valittuna on emulaattori, voidaan koodi ajaa painamalla F5. Tämän jälkeen Visual Studio käynnistää Windows Phone -emulaattorin, ja ajaa siinä valitun projektin. Sovelluksen ollessa käynnissä voidaan sitä testikäyttää haluamalla tavalla (kuva 11).

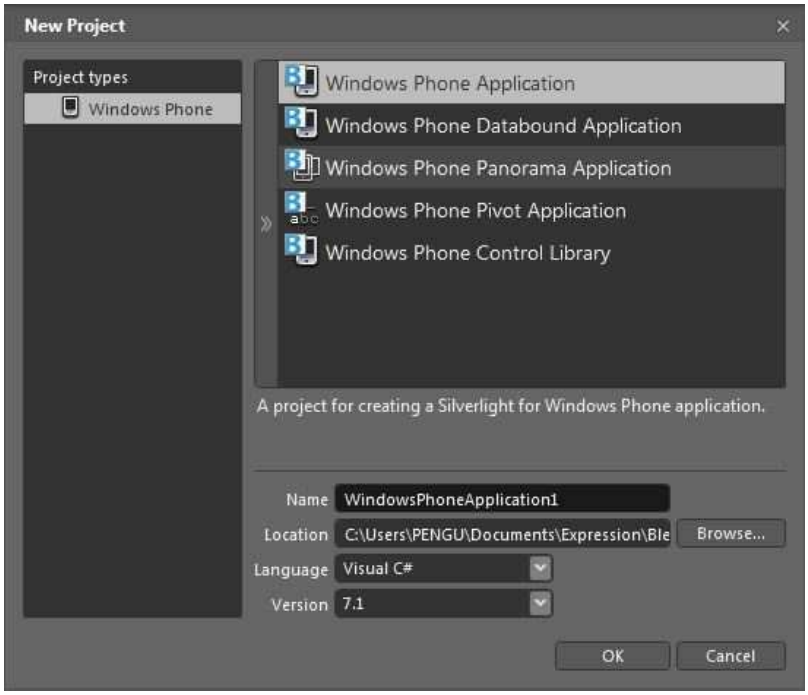
Kuva 11. Windows Phone -emulaattori ja testisovellus.

Testikäytön jälkeen on emulaattori suositeltavaa jättää käyntiin. Sovellus pitää kuitenkin sulkea ennen kuin sen ohjelmointia voidaan jatkaa. Tämä tapahtuu painamalla emulaattorissa olevaa paluunäppäintä niin kauan, kunnes sovellus

on sulkeutunut. Tämän jälkeen Visual Studio vapauttaa editorin taas kehittäjän käyttöön. Emulaattorin käyttöön ja mahdollisuuksiin tutustumme tarkemmin työn luvussa 2.4.

2.3 Expression Blend 4

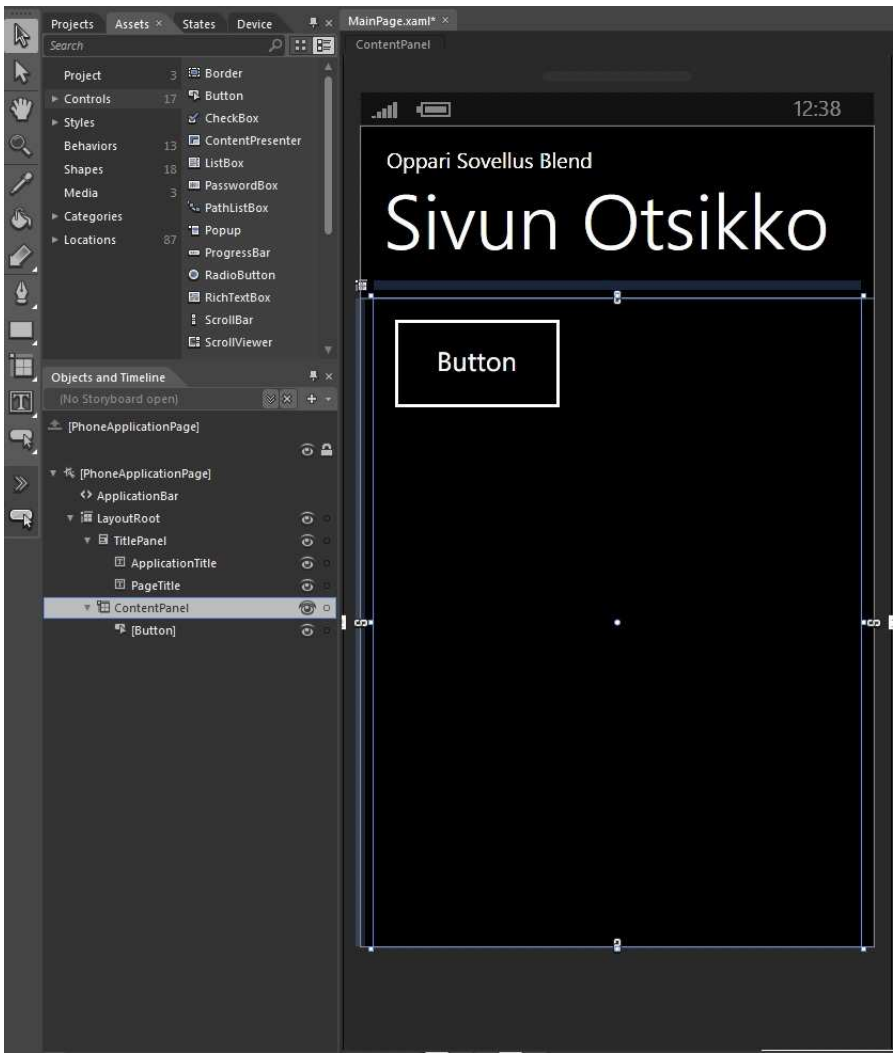
Ohjelmistokehittäjän halutessa luoda graafisesti näyttäviä, interaktiivisia Silverlight-sovelluksia, on hänen syytä ottaa käyttöönsä Visual Studion sijasta Expression Blend niminen työkalu. Työkalu tulee Windows Phone SDK -paketin mukana, ja se on Visual Studiosta erillään oleva kehitystyökalu. Expression Blend -työkalua ei kuitenkaan ole tarkoitettu ohjelmakoodin kirjoittamiseen, vaan itse ohjelmointityö on parempi tehdä Visual Studiolla. Expression Blendiä voidaan kuitenkin käyttää vaikka ohjelmiston kehitys olisi aloitettu aiemmin Visual Studiolla. Projekti voidaan näin ollen avata kesken ohjelmoinnin myös Expression Blend -työkalulla ja lähteä muokkaamaan sitä halutulla tavalla, esimerkiksi animaatioida luoden. Myös uuden Silverlight -projektin aloittaminen suoraan Expression Blendiin on mahdollista (kuva 12).



Kuva 12. Uuden projektin luonti Expression Blend 4 -työkalulla.

Visual Studion tapaan myös Expression Blend sisältää eri osiot, joita kehittäjä voi käyttää työssään. Blendissä näitä osioita on kaksi, Design työtila ja Animation työtila. Erona Visual Studioon on se, että vain yksi osio on kerrallaan kehittäjän nähtävissä. Työtiloja voidaan vaihtaa painamalla F6-näppäintä. Molemmilla työtiloilla on oma käyttötarkoituksensa, joka selviää kehittäjälle jo niiden nimistä. Design työtilaa käytetään käyttöliittymäsuunnitteluun, kun taas Animation työtilaa käytetään animaatioiden luomiseen.

Design työtila sisältää samat kontrollit kuin Visual Studio, ja niiden lisääminen projektiin toimii vastaavalla tavalla. Design työtilassa kontrollit löytyvät vasemmasta reunasta Assets nimisestä valikosta (kuva 13).

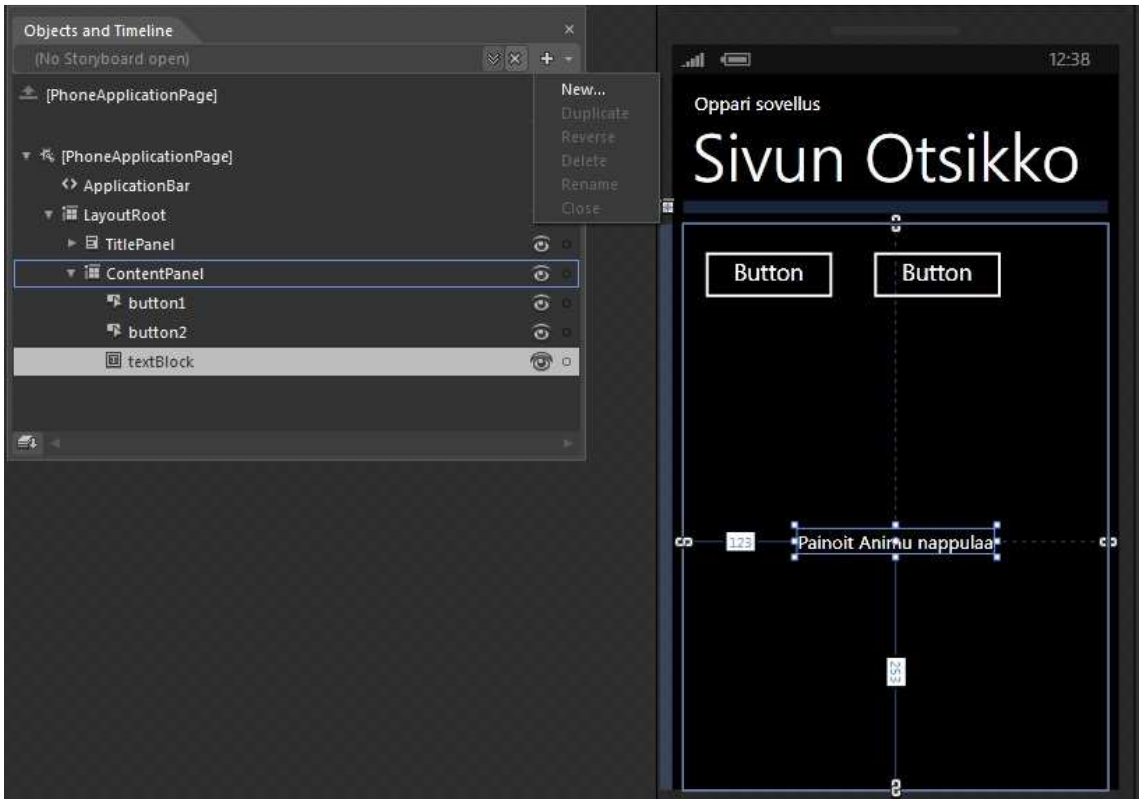


Kuva 13. Design työtila.

Kun projektin sivulle on lisätty haluttuja kontrolleja, voidaan niiden tarkempia ominaisuuksia tarkastella työtilan oikeasta reunasta Properties -valikosta (kuva 14). Blend tarjoaa runsaan määrän ominaisuuksia eri kontrolleille, ja näin ollen kehittäjällä on mahdollisuus muokata kontrollien visuaalisuutta runsaammin.

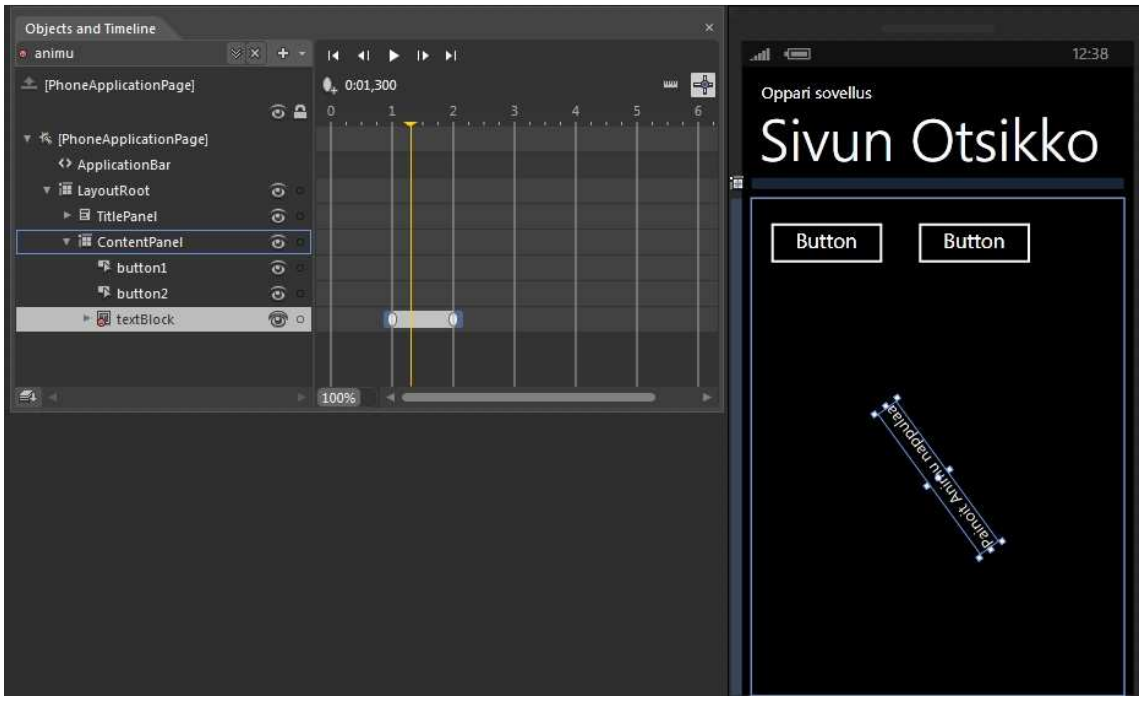
Kuva 14. Kontrollien ominaisuuksia Blendissä.

Animaatioiden luonti Expression Blendissä on yksinkertaista. Animaatioita voidaan luoda esimerkiksi projektiin lisätyille kontrolleille. Animaatio luodaan luonnollisesti Animation työtilassa. Animaatio työtilaan siirtymisen jälkeen voidaan kontrolleja animoida Objects and Timeline nimisen paneelin aikajanan avulla. Animoitava kohde valitaan hiiren painikkeella, jonka jälkeen luodaan siihen uusi animaation aikajana Objects and Timeline paneelista valitsemalla New (kuva 15). Tämän jälkeen Expression Blend käskee antamaan animaatiolle nimen.



Kuva 15. Kontrollin uuden animaation luonti.

Kun uusi animaatio on luotu, siirtyy Expression Blend ns. äänitystilaan. Äänitystila ilmenee punaisista reunuksista työtilan ympärillä. Aikajanaa käyttämällä voidaan animoitavalle kontrollille luoda vaikkapa jokin liike-efekti, kuten pyörähdys tai liukuva liike sivun reunalta reunalle. Aikajanalta valitaan ensin sekuntikohta, jossa animaatio halutaan päättyvän, ja samalla siirretään valittu kontrolli kohtaan, johon sen halutaan siirtyvän animaation aikana (kuva 16). Näiden tietojen perusteella Expression Blend luo automaattisesti animaation kontrollin alku- ja loppupisteen välille. Kun animaatio on luotu voidaan äänitys sulkea samasta paikasta kuin se aloitettiin valitsemalla Close.



Kuva 16. Animointiliikkeen luominen aikajanan avulla.

Kun animaatio on luotu, on se mahdollista laukaista erilaisilla menetelmillä. Yksinkertaisin tapa on laittaa animaatio käynnistymään napin painalluksella. Tämä voidaan tehdä kirjoittamalla käynnistysfunktio jonkin näppäimen ohjelmakoodiin (kuva 17). Koodiin päästään Expression Blendissä käsiksi avaamalla sivun .cs -tiedosto Design työtilan vasemmasta reunasta, kohdasta Projects. Blendissä editori toimii vastaavalla tavalla kuin Visual Studiossa.

```
private void button1_Click(object sender, RoutedEventArgs e)
{
    MessageBox.Show("Painoit sovelluksessa näkyvää nappulaa.");
}

private void button2_Click(object sender, RoutedEventArgs e)
{
    animu.Begin();
}
```

Kuva 17. Animaation käynnistyksen ohjelmakoodi.

Kun haluttu sovellus on saatu valmiiksi, pitää se rakentaa ja testiajaa emulaattorissa. Expression Blendissä rakennus- ja virheenjäljitystoiminnot toimivat sa-

moin kuin Visual Studiossa. Ainoana erona on projektin rakentamisen pikanäppäin, joka Blendissä on näppäinyhdistelmä Ctrl+Shift+B.

2.4 Windows Phone -emulaattorin ominaisuudet

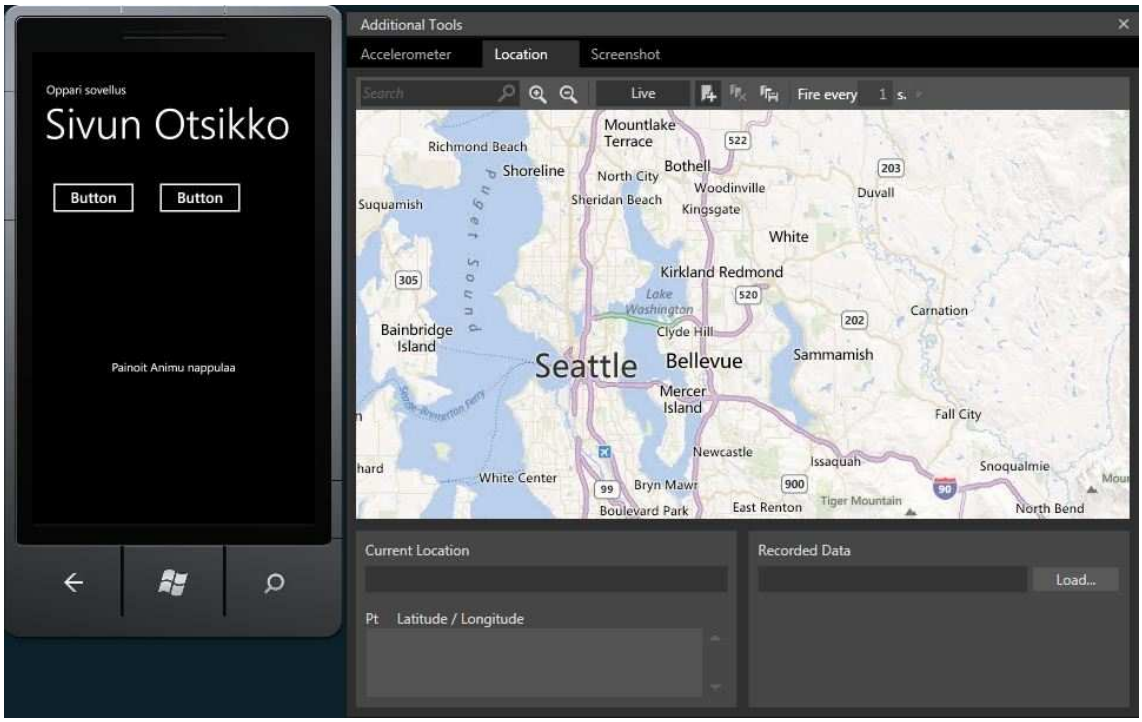
Kehitystyökalujen mukana tulevassa Windows Phone -emulaattorissa tarjotaan sisäänrakennettuna suurin osa niistä ominaisuuksista, joita itse älypuhelimessakin on. Tällä mahdollistetaan se, että kehittäjän ei välttämättä tarvitse omistaa omaa Windows Phone -älypuhelinta testatakseen tekemiensä ohjelmien toimivuutta. Emulaattoriin on myös rakennettu joitain sellaisia ominaisuuksia, joita varsinaisista puhelimista ei löydy. Emulaattorilla on mahdollista esimerkiksi ottaa ruudunkaappauksia testattavista sovelluksista. Näitä ruudunkaappauksia tarvitaankin, kun sovellusta julkaistaan sovelluskauppaan.

Emulaattori ei kuitenkaan ole täysin puutteeton. Suurimpana puutteena tällä hetkellä on puhelinverkon puuttuminen. Tämä tarkoittaa sitä, että sovelluskehittäjä ei voi testata emulaattorilla sovelluksen sellaisia ominaisuuksia, jotka liittyvät tekstiviesteihin ja puheluihin. Emulaattoriin on myös ainoastaan sisällytetty kaksi sensoritoimintoa. Näitä sensoreita ovat kiihtyvyys/orientaatio-sensori ja paikkatietosensori. [1, s. 58]

Emulaattorissa itsessään on kuvan 18 mukainen työkalurivi, josta pääsee käyttämään eri lisätoimintoja.

Kuva 18. Emulaattorin työkalurivi lisätoiminnoille.

Työkalurivistä käsin käyttäjän on mahdollista vaihdella emulaattorin asentoa pysty- ja vaaka-asennon välillä, vaihtaan tarkennustilaa (zoom) sekä avata lisätoimintojen erillinen valikko. Lisätoimintojen valikosta pääsee käsiksi sensoreihin ja ruudunkaappaustoimintoon (kuva 19).



Kuva 19. Windows Phone -emulaattorin lisävalikko.

Ensimmäinen välilehti lisävalikossa on varattu kiihtyvyys/orientaatio-sensorille. Tästä osiosta voidaan vaihdella halutulla tavalla puhelimen kallistuskulmia ja asentoa, tai vaihtoehtoisesti ravistaa puhelinta automatisoidulla toiminnolla.

Toinen välilehti on tarkoitettu paikkatietosensoriin liittyvien toimintojen testaamiseen (kuva 19). Karttana sijaintitietojen testaamisessa on käytössä Microsoftin Bing-kartat. Karttaa voidaan liikutella hiiren avulla haluttuun paikkaan tai käyttää hakutoimintoa jonkin tietyn paikan etsimiseen. Karttaan voidaan myös lisätä kaksoisklikkaamalla eri pisteitä ja näiden avulla muodostaa polku, joka sitten voidaan ajaa painamalla valikon play -näppäintä. Tämä simuloi kännykän liikettä valittua polkua pitkin. Myös luotujen polkujen tallennus onnistuu valikosta löytyvällä tallennuspainikkeella.

Viimeiseltä välilehdeltä löytyy ruudunkaappaustoiminto (kuva 19). Tämä toiminto ottaa napin painalluksella ruudunkaappauksen emulaattorin ruudusta ja siinä sillä hetkellä näkyvästä näkymästä. Ruudunkaappaukset on hyvä ottaa siten, että zoom-asetus on määritelty 100 %:iin, koska muuten kuvat voivat olla suttuisia.

3 WINDOWS PHONE MARKETPLACE JA SOVELLUKSEN JULKAISU

Älypuhelinmarkkinoiden kilpailun kovetessa yhtenä tärkeimmistä asioista voidaan pitää sovelluskauppaa. Tämä on myös kehittäjän kannalta oleellinen asia, sillä tehdyt sovellukset julkaistaan kunkin mobiilikäyttöjärjestelmän omassa sovelluskaupassa. Microsoftin Windows Phone -mobiilikäyttöjärjestelmä tarjoaa sovellusten kauppapaikaksi Windows Marketplacen.

Windows Marketplace on suljettu sovelluskauppa. Tämä tarkoittaa sitä, että kehittäjän ainut mahdollisuus sovellusten julkaisuun on Windows Marketplacen kautta. Kehittäjä ei myöskään voi ajaa sovelluksia omassa puhelimessaan ennen kuin hän on rekisteröitynyt sovelluskauppaa, ja tämän avulla poistanut puhelimen lukituksen. Sovelluskaupan sulkeutuneisuus helpottaa Microsoftin sovelluksiin kohdistuvaa valvontaa. Tämän avulla taataan sovelluksille parempi laatu sekä tietoturva.

Älypuhelimen käyttäjä pääsee käsiksi sovelluskauppaan ja sieltä ladattavissa oleviin sovelluksiin suoraan puhelimensa kautta. Puhelimen aloitusnäkymä sisältää vakiona tiilen nimeltä Marketplace, joka painalluksen tapahtuessa avaa Windows Marketplacen. Sovelluskaupan sovellukset avautuvat käyttäjän eteen genreittäin. Tiettyä genreä painamalla pääsee käyttäjä selaamaan juuri siihen genreen kuuluvia sovelluksia. Sovellukset voivat olla kehittäjän asetusten mukaan joko ilmaisia tai maksullisia.

Marketplacen sovellustarjonta on Forbesin internetsivuilla toukokuussa 2012 kirjoitetun artikkelin mukaan ylittänyt 100 000 sovellusta. Sovelluskaupan tarjonta kasvaa tämän lisäksi päivittäin yli 300 sovelluksella. [16]

3.1 Sovelluskauppaan rekisteröityminen

Ennen sovellusten julkaisua on sovelluskehittäjän rekisteröitävä itsensä Windows Phone Dev Centerin käyttäjäksi. Tähän sovelluskehittäjä tarvitsee ensiksi

Windows Live -tunnukset. Live -tunnusten luonti onnistuu internetissä osoitteessa www.create.msdn.com. Kun tunnukset on luotu voi sovelluskehittäjä kirjautua sisään Dev Center -palveluun. Vasta tämän jälkeen hänellä on mahdollisuus suorittaa maksun vaativa rekisteröinti, joka vaaditaan sovelluksen julkaisua varten. Microsoft on hinnoitellut sovelluskauppaan liittymisen 99 dollarin eli noin 75 euron hintaiseksi per vuosi. Rekisteröintimaksu voidaan suorittaa joko luottokortilla, PayPal tilillä tai vaihtoehtoisesti saadulla markkinointikoodilla. Microsoft antaa kehittäjälle mahdollisuuden liittyä sovelluskauppaan yksityshenkilönä tai opiskelijana, mutta myös yrityksille on oma rekisteröitymismahdollisuutensa. [17]

3.2 Puhelimen lukituksen poisto

Kun kehittäjä on maksanut vuotuisen rekisteröitymismaksun, on hänellä mahdollisuus poistaa omasta puhelimestaan lukitus. Sovelluskehittäjän on poistettava puhelimen lukitus, mikäli hän haluaa testata kirjoittamiaan sovelluksia puhelimessaan.

Lukitus poistetaan Visual Studio 2010 Expressin mukana tulleella erillisellä ohjelmalla nimeltä PhoneReg.exe. Sovellus löytyy Windows Phone SDK:n luomasta asennuskansiosta. Puhelin kytketään tietokoneeseen USB-kaapelilla, jonka jälkeen PhoneReg ohjelma ajetaan tietokoneella. Ohjelman suorittamisen jälkeen puhelimen lukitus on poistettu käytöstä.

Kun lukitus on poistettu pitää kehittäjän enää asentaa Microsoft Zune -ohjelmisto, joka niin ikään vaaditaan sovelluksen ajamiseen puhelimessa. Jos Zune -ohjelmistoa ei ole asennettu, antaa Visual Studion virheenjäljitystoiminto kuvan 20 mukaisen virheilmoituksen.

Kuva 20. Virheenjäljitystoiminto ilmoittaa Zunen puuttumisesta.

Zune on Microsoftin video- ja musiikkipalvelu. Se on ladattavissa ilmaiseksi osoitteesta www.zune.net. Kun Zune -palvelu on asennettu ja puhelimen lukitus poistettu onnistuneesti, voi sovelluskehittäjä ajaa tekemänsä sovelluksen omassa puhelimessaan.

3.3 Sovelluskauppa tulon lähteenä

Microsoft tarjoaa sovelluskehittäjille mahdollisuuden ansaita rahaa tekemillään sovelluksilla. Mahdollisia ansaintatyylejä on kaksi: joko sovellusten asettaminen maksulliseksi, tai sitten niin, että sovellukset sisältävät tilan mainosbannerille. Molemmissa tapauksissa sovelluksen on kuitenkin saatava käyttäjiä, ennen kuin se voi tuottaa kehittäjälleen rahaa.

Tulonjako

Sovelluskauppa ei tarjoa kehittäjälle täyttä tuloa myydyistä sovelluksista. Kaikista sovelluskaupasta myydyistä sovelluksista ottaa Microsoft provision. Tulonjakosuhde kehittäjän ja Microsoftin välillä on 70/30. Tämä tarkoittaa siis sitä, että kehittäjä saa 70 % sovelluksensa tuotoista, kun taas Microsoft loput.

Tämä ei kuitenkaan kata kaikkia tuloista tehtäviä vähennyksiä. Microsoftin provision lisäksi jokaisesti myydystä sovelluksesta vähennetään kunkin kohdemaan verot. Jos otetaan sovelluskaupan yleisin sovelluksen myyntihinta, 0,99

euroa, ja lasketaan tästä pois verot ja provisio, saadaan sovelluskehittäjän ansioksi 0,56 euroa.

Myös sovelluksen käyttöliittymässä näkyvät mahdolliset mainokset voivat tuoda kehittäjälle tuloja. Mainoksia sisältävät sovellukset tarjotaan yleensä kuluttajalle ilmaiseksi, mutta kehittäjä saa osansa mainostuloista. Mainostulojen määrä riippuu mainosten katselukerrasta, eli toisin sanoen siitä, kuinka usein sovellusta käytetään ja kuinka moni käyttää sitä. Mainokset voidaan lisätä sovellukseen käyttäen Microsoftin AdControl -kontrollia.

3.4 Sovelluksen julkaisu ja sertifiointi

Valmiit sovellukset julkaistaan Windows Phone Dev Center -internetsivuilla olevalla valmiilla toiminnolla. Tämä on tehtävä samalla tietokoneella, jossa sovellus on rakennettu Visual Studiota käyttäen.

Ennen kuin sovelluskehittäjä saa sovelluksensa näkymään sovelluskaupassa, on sen käytävä läpi Microsoftin sertifiointi. Sertifiointi tapahtuu, kun kehittäjä on saanut sovelluksensa valmiiksi ja lähettää siitä Visual Studiolla luodun .xap -tiedoston sovelluskauppaan. Tämä xap-tiedosto sisältää koko tehdyn sovelluksen ja projektiin liittyvät tiedostot.

Sovelluksen lähettäminen on 5-vaiheinen ohjattu prosessi, jossa esitellään kaikki julkaisuun tarvittavat toiminnot. Ohjatuissa vaiheissa kehittäjä nimeää oman sovelluksensa, antaa sille kuvauksen, toimittaa tarvittavat kuvamateriaalit, päättää maat joihin sovellus julkaistaan ja hinnoittelee sovelluksensa. Viimeisessä vaiheessa prosessi lähettää sovelluksen eteenpäin tarkistusta varten. [18]

Prosessin vaiheista suurimmat ongelmat aiheutuvat yleisesti kuvamateriaalien toimittamisesta. Microsoft vaatii sovellukseen seuraavat kuvamateriaalit:

- suuri aloitusnäkymän tiili (173x173 pikseliä)
- pieni valikon tiili (99x99 pikseliä)
- suuri PC-alustan tiili (200x200 pikseliä)

- vähintään yksi ruudunkaappaus sovelluksesta [18].

Kehittäjä joutuu näistä kuvamateriaaleista luomaan kolme ensimmäistä itse oman makunsa mukaan. Sovelluksen ruudunkaappaukset sen sijaan ovat helposti otettavissa emulaattorin avulla.

Sovelluksen sertifiointitarkastuksen kestoksi Microsoft lupaa keskimääräisesti 3-5 päivää. Jos kehittäjä on panostanut sertifiointivaatimuksiin, saattaa sovellus läpäistä tarkastuksen ensimmäisellä kerralla. Jos näin ei tapahdu, antaa Microsoft kehittäjälle raportin korjausta vaativista asioista. Sertifioinnin tekniset vaatimukset löytyvät listattuna Dev Center -sivustolta hakusanalla technical certification requirements. [1, s. 219]

Kun koko prosessi on onnistuneesti läpäisty, ilmestyy sovellus ladattavaksi sovelluskauppaan. Sovelluksen voivat tämän jälkeen ladata sovelluskaupasta kaikki Windows Phone 7 -älypuhelinten omistajat.

4 ONTWITTER-SOVELLUKSEN TOTEUTUS

Tässä opinnäytetyössä esiteltävä Windows Phone 7 -alustalle luotu puhelinsovellus kehitettiin vain ja ainoastaan mielenkiinnosta alustaa kohtaan. Sovellusta ei tulla julkaisemaan Microsoftin sovelluskaupassa. Sovelluksen tarkoituksena on toimia esimerkkinä sille, kuinka helposti sovelluskehitys WP7 -alustalle on omaksuttavissa. Sovelluksessa hyödynnetään myös alustaan liittyviä hyvin keskeisiä tekniikoita.

4.1 Twitter yhteisö- ja mikroblogipalvelu

Twitter on vuonna 2006 perustettu reaaliaikainen yhteisö- ja mikroblogipalvelu. Twitter on lähtöisin San Franciscosta, mutta nykyään sitä käytetään lähes jokaisessa maailman maassa, ja se on tarjolla yli 20 eri kielellä.

Twitterin pääasiallinen tarkoitus on antaa sen käyttäjille mahdollisuus jakaa toisilleen viestejä. Näitä viestejä kutsutaan ”twiiteiksi”. Viestien maksimipituudeksi on määritetty 140 merkkiä. Palvelussa olevat twiitit ovat nähtävissä kaikille palvelun käyttäjille myös ilman rekisteröitymistä. [19]

Rekisteröityessään palveluun saa käyttäjä mahdollisuuden seurata muita rekisteröityneitä käyttäjiä. Näiden seurattujen käyttäjien lähettämät twiitit näkyvät suoraan kunkin seuraajan aikajanalla (timeline). Samalla rekisteröityneelle käyttäjälle annetaan mahdollisuus julkaista omia twiittejään, joita muut voivat seurata vastaavalla tavalla.

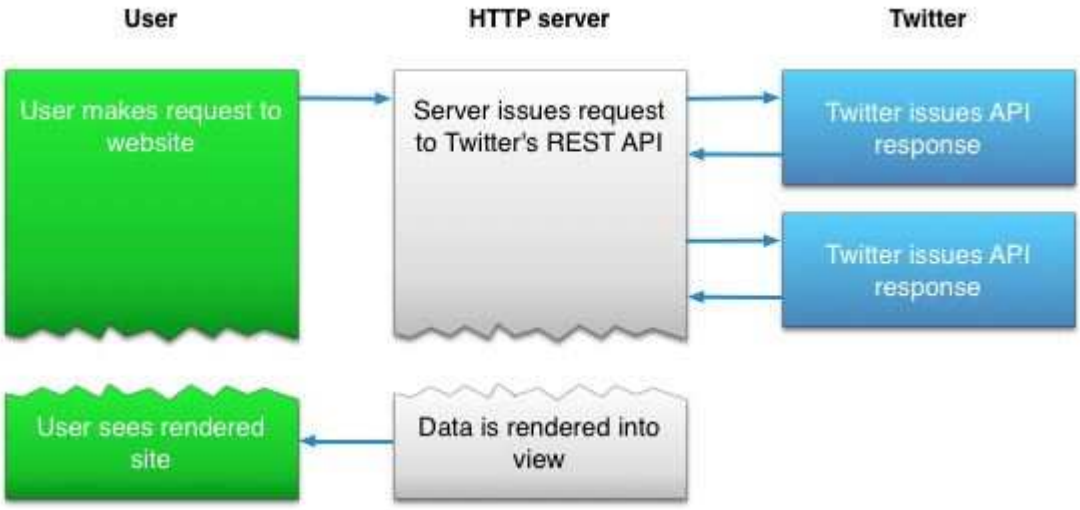
4.2 Ohjelmointirajapinta ja Twitter API

Ohjelmointirajapinnan (API) tarkoituksena on luoda kehittäjälle mahdollisuus käyttää jonkun ohjelman yleisimpiä toimintoja. Ohjelmointirajapinta on siis ikään kuin välikäsi, jota sovelluskehittäjän luoma sovellus voi käyttää kommunikointiin jonkin toisen ohjelman kanssa. Tällä vältetään myös kolmannen osapuolen so-

velluksen puuttuminen suoraan rajapinnan alla oleviin menetelmiin. Kun ohjelmointirajapinta on käytössä jossain verkkopalvelussa käytetään siitä yleisesti nimeä Web-API. [20]

Myös Twitter-yhteisöpalvelulla on oma ohjelmointirajapintansa, kuten suurimmalla osalla yleisimmistä sosiaalisen median palveluista. Twitter tarjoaa kolmannen osapuolten sovellusten kehittäjille kaksi erillistä API:a: REST API:n ja Streaming API:n.

Twitterin REST (Representational State Transfer) API pitää sisällään yleisimmät twitterin toiminnallisuudet. Tämän rajapinnan avulla sovellus ottaa yhteyden twitterin toimintoihin ja käyttää niitä. REST API:a käytettäessä sovellusten välinen yhteys avataan vain tarvittaessa. Kun tarvitut tiedot on saatu Twitter-palvelusta, palautetaan vastausdata kyselyn tehneelle käyttäjälle. Kyselyn tekeminen on esitelty kuvassa 21. [21]



Kuva 21. Kysely Twitterin REST API:a käyttäen. [21]

Streaming API:lla muodostetaan jatkuvasti avoinna oleva HTTP-yhteys Twitterin rajapintaan. Tämän avulla mahdollistetaan mm. reaaliaikainen viestien välitys sovelluksen ja twitterin välillä. [21]

4.3 ONTwitter-sovelluksen käyttöliittymä

Käyttöliittymän suunnittelussa haluttiin keskittyä kahteen pääasiaan: yksinkertaisuuteen ja akunkestoon. Näiden perusteella Visual Studio projektin pohjaksi valittiin yhden sivun projekti. Tällä saatiin heti alussa poistettua sovelluksesta kaikki ylimääräiset akkua kuluttavat animaatiot. Myöskin yhden sivun sovelluksen ajateltiin tuovan sovellukseen haluttua yksinkertaisuutta. Kaikki käyttäjän tarvitsema päätieto haluttiin tuoda näkyviin yhdelle oletussivulle.

Pääsivun käyttöliittymä on jaettu suunnittelussa kolmeen osaan. Osat ovat luotu Microsoftin mallipohjaan valmiiksi luotuihin kahteen osaan. Tämän ratkaisun avulla sovelluksen käyttöliittymän ulkoasu pystyttiin säilyttämään yhtenäisenä puhelimeen integroitujen sovellusten ulkoasun kanssa. Nämä kolme pääosaa on esitelty kuvassa 22.

Kuva 22. Käyttöliittymän etusivun kolme pääosaa.

Pääsivun ensimmäinen osa koostuu otsikkorivistä. Pienemmällä fontilla luotu otsikko on pohjaan valmiiksi määritelty sovelluksen otsikko. Seuraava isommalla fontilla näkyvä otsikko pitää sisällään käyttöliittymän sivun varsinaisen otsikon.

Toinen osa pääsivulla pitää sisällään toiminnon, jota käyttäjä tarvitsee sovellusta käyttääkseen. Tämä osio tarjoaa käyttäjälle kirjoituskentän ja painikkeen, johon voidaan ohjelmoida tarvittavat toiminnot.

Kolmannessa osassa on käytössä Visual Studion toolboxista valittu kontrolli nimeltä ListBox. Tähän kontrolliin voidaan listata sellaista dataa jota sovelluksen halutaan näyttävän.

4.4 Värimaailman suunnittelu

Käyttöliittymän värejä valittaessa oli alkuperäisenä tarkoituksena valita värejä, jotka ovat käytössä myös Twitter-palvelussa. Alkuperäisestä suunnitelmasta poiketen koko värimaailma kuitenkin päätettiin luoda itse. Värimaailmassa käytettiin pääasiassa hieman tummempia värejä akkukeston parantamiseksi. Mitä tummempia värejä käyttöliittymässä käytetään, sitä vähemmän valotehoa mobiililaitteen näyttö joutuu tarjoamaan. Käytettävän valotehon määrällä on suora vaikutus laitteen virrankulutukseen ja näin ollen akun kestoon.

Värimaailman suunnittelun jälkeen halutut värit asetettiin sovellukseen XAML kieltä käyttäen. Samalla toolboxista lisätyille kontrolleille tehtiin viimeiset asettelumuutokset. Asettelumuutokset tehtiin myös käyttäen XAML-koodieditoria. Valmis XAML-koodi sisältää seuraavanlaiset määritykset eri käyttöliittymän etusivun pääosille:

Sovelluksen tausta ja väri:

```
<Grid x:Name="LayoutRoot" Background="#FF0F2648">
```

Sovelluksen etusivun pääosa 1:

```
<StackPanel x:Name="TitlePanel" Grid.Row="0" Mar-
gin="12,17,0,28">
<TextBlock x:Name="ApplicationTitle" Text="ONTwitter"
Style="{StaticResource PhoneTextNormalStyle}"/>

<TextBlock x:Name="PageTitle" Text="Search" Margin="9,-7,0,0"
Style="{StaticResource PhoneTextTitle1Style}" Fore-
ground="#FF00FCFF" />
</StackPanel>
```

Sovelluksen etusivun pääosa 2:

```
<Button Content="Refresh" Height="72" HorizontalAlignment="Left"
Margin="290,6,0,0" Name="btnRefresh" VerticalAlignment="Top"
Width="160" Grid.RowSpan="2" Click="btnRefresh_Click" Border-
Brush="#FF00FDFB" BorderThickness="1" Background="#FFBCBCBC"
Foreground="Black" FontWeight="ExtraLight" FontSize="18" />

<TextBox Height="72" HorizontalAlignment="Left" Margin="0,6,0,0"
Name="txtName" Text="" VerticalAlignment="Top" Width="296"
Grid.RowSpan="2" FontSize="22" FontStretch="Normal" Font-
Weight="Light" BorderThickness="1" BorderBrush="#FF00FCFF" />
```

Sovelluksen etusivun pääosa 3:

```
<ListBox Grid.Row="1" HorizontalAlignment="Left" Mar-
gin="6,33,0,6" Name="lstTwiitit" Width="444">
<ListBox.ItemTemplate>
<DataTemplate>

<StackPanel Orientation="Horizontal" Height="110" Margin="-10,-
10,-10,-10">

<Image Source="{Binding ImageSource}" Height="73" Width="73"
VerticalAlignment="Top" Margin="10,10,8,8" />

<TextBlock Text="{Binding Message}" Margin="1,10,10,10"
TextWrapping="Wrap" FontSize="16" Width="350" Font-
Weight="Normal" FontFamily="Segoe WP Semibold" />

</StackPanel>
</DataTemplate>
</ListBox.ItemTemplate>
</ListBox>
```

Lisättyjen kontrollien kohdalla XAML-koodissa on hyvä huomioida, että kontrollit on nimetty yleisesti käytetyn nimeämiskäytännön mukaan. Tämä tarkoittaa sitä, että kontrollien nimen etuliite määräytyy käytössä olevan kontrollin mukaan. Esimerkiksi TextBox kontrollin nimeämisessä käytetty etuliite on ”txt”. Vastaavasti ListBox kontrollin nimen etuliite on ”lst”, ja Button kontrollin etuliite on ”btn”.

Kun koodi on valmis, näyttää Visual Studion esikatseluruutu millaiselta sovelluksen etusivu näyttää. Sovelluksen pääsivun valmis ulkoasu XAML-koodin kirjoittamisen jälkeen on nähtävissä kuvassa 23.

Kuva 23. Sovelluksen valmis pääsivu muotoiluineen ja väreineen.

4.5 Toiminnallisuuksien suunnittelu ja toteutus sovellukseen

Sovelluksen pääasiallinen tarkoitus on mahdollistaa käyttäjälle Twitter-palvelun jonkun käyttäjän viestien eli twiittien lukeminen. Tätä toiminnallisuutta varten sovellukseen oli suunniteltava toiminto, joka ottaa yhteyden Twitterin ohjelmointirajapintaan eli API:in. Kuten aiemmin työssä esiteltiin, tarjoaa palvelu kaksi mahdollista rajapintaa sovelluksien väliseen kommunikointiin. Twitter käyttäjien twiitteihin pääsee käsiksi REST API rajapinnan kautta.

4.5.1 Yhteyden muodostaminen Twitter-palveluun

Yhteyden muodostaminen REST API rajapintaan voidaan tehdä käyttämällä C#-ohjelmointikielen WebClient nimistä luokkaa. WebClient -luokan avulla voi-

daan yhdistää suoraan jonkin web-palvelun osoitteeseen. Yhdistäminen onnistuu seuraavalla koodilla:

```
private void btnRefresh_Click(object sender, RoutedEventArgs e)
{

WebClient twitter = new WebClient();
twitter.DownloadStringCompleted += new    DownloadStringCom-
pletedEventHandler(twitter_DownloadStringCompleted);

twitter.DownloadStringAsync(new
Uri("http://api.twitter.com/1/statuses/user_timeline.xml?screen_
name=" + txtName.Text));

}
```

Sovelluksessa yhdistämistoiminto on asetettu toimimaan aiemmin XAML-koodilla luodun ”btnRefresh” -painikkeen painalluksella. Kun painiketta painetaan, luodaan ohjelmakoodissa uusi twitter niminen WebClient. Tämä taas ottaa asynkronisesti yhteyden määritettyyn osoitteeseen. Osoitteeksi on tässä tapauksessa valittu Twitter REST API:n ohjeistama osoite, josta löytyy Twitter käyttäjän aikajana eli timeline. Lopuksi osoitteen perään annetaan käyttäjänimi hakusanana. Käyttäjänimi tieto joka kyselyssä ajetaan saadaan etusivulle lisätystä TextBox kontrollista, jonka nimeksi oli määritetty ”txtName”. Kun kysely on välitetty WebClient luokan avulla API:lle, palauttaa API tämän jälkeen kyselyn tulokset xml -tiedostossa.

4.5.2 Kyselyn tulosten parsiminen

Kun ohjelmointirajapinta on palauttanut tehdyn kyselyn tulokset, pitää tuodut tiedot jäsentää eli parsia. Kun palautettu kysely parsitaan saadaan sieltä eroteltua se data, joka halutaan näyttää sovelluksen käyttäjälle. Tässä tapauksessa palautetusta syötteestä halutaan parsia esille varsinainen viesti eli twiitti sekä twiittaajan tekijän asettama profiilikuva. Nämä tiedot halutaan näyttää etusivulle määritetyssä ListBox kontrollissa listana.

Halutut tiedot saadaan näkyviin ListBox kontrollissa käyttämällä Data Binding nimistä ominaisuutta. Tällä ominaisuudella mahdollistetaan se, että haetut tiedot saadaan näkyviin käyttöliittymään ilman erillistä näkymän päivitystä. Data Binding ListBox kontrollissa toimii seuraavalla XAML-koodilla:

```
<Image Source="{Binding ImageSource}" Height="73" Width="73"
VerticalAlignment="Top" Margin="10,10,8,8" />

<TextBlock Text="{Binding Message}" Margin="1,10,10,10"
TextWrapping="Wrap" FontSize="16" Width="350" Font-
Weight="Normal" FontFamily="Segoe WP Semibold" />
```

ListBox kontrollin sisällä olevaan kuvalähteeseen asetetaan Binding Source nimeltä “ImageSource”. TextBlock kontrollin tekstikenttään taas määritetään Binding Source nimeltä ”Message”.

Binding Source tietojen määrittämisen jälkeen on projektiin luotava uusi luokka. Projektiin lisättyyn luokkaan määritellään Binding Sourceja vastaavat määritelmät, jotka määrittävät ja saavat arvoja. Lisätyn luokan koodi näyttää seuraavalta:

```
namespace PhoneApp2
{
          public class Twitter
          {
                    public string ImageSource { get; set; }
                    public string Message { get; set; }
          }
}
```

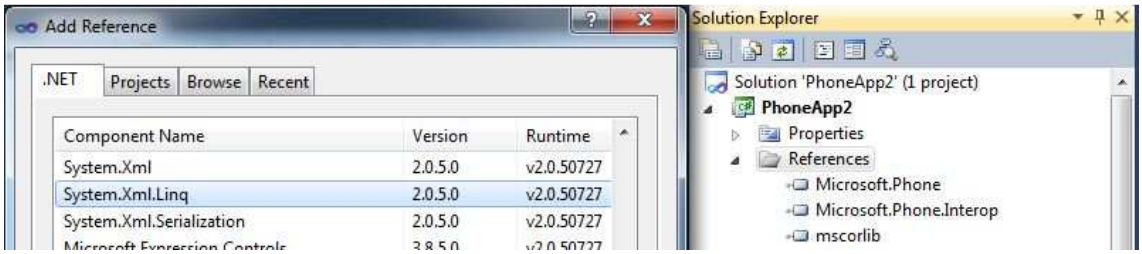
Kun luokka on luotu, parsitaan API:lta saadun kyselyt tulokset ja syötetään ne luodun luokan propertyihin. Parsiminen tapahtuu C#-koodilla:

```
void twitter_DownloadStringCompleted(object sender, Down-
loadStringCompletedEventArgs e)
{

XElement xmlTweets = XElement.Parse(e.Result);

lstTwiitit.ItemsSource = from tweet in xmlT-
weets.Descendants("status")
```

Tiedon parsimisen hoitaa XElement nimisen luokan toiminto. Jotta XElement luokkaa voidaan käyttää sovelluksessa, täytyy sen referenssi lisätä projektille. XElement luokan vaativa referenssi on nimeltään System.Xml.Linq. Referenssin voi lisätä Solution Explorerin kautta Add Reference toiminnolla. Lisäys on esitelty kuvassa 24.



Kuva 24. System.Xml.Linq referenssin lisääminen projektiin.

Kun tieto on parsittu, syötetään siitä halutut tiedot aiemmin määriteltyihin Binding Sourceihin. Tämä onnistuu seuraavasti:

```
select new Twitter
{
ImageSource =
tweet.Element("user").Element("profile_image_url").Value,
Message = tweet.Element("text").Value

};
```

Kun halutut arvot on syötetty, tulostuvat ne automaattisesti pääsivun ListBox kontrollin sisälle kun ohjelman toimintoa on käytetty. Onnistuneesti suoritettu käyttötesti nähtävillä kuvassa 25.

Kuva 25. Twiittien hakutoiminnon suoritus ONTwitter-sovelluksella.

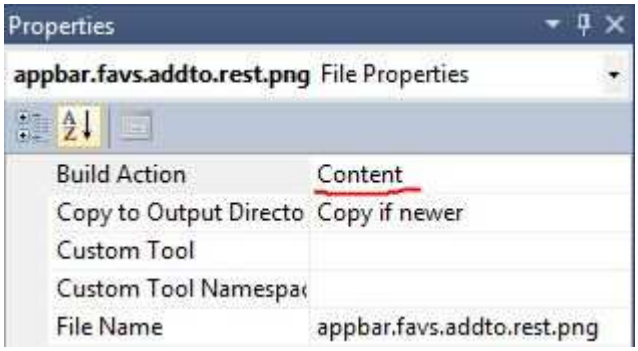
4.6 Sovelluksen interaktiivisuus ja käyttökokemuksen rikastuttaminen

4.6.1 Application Bar -valikko

Sovelluksen käyttökokemuksen parantamiseksi sovellukseen suunniteltiin pääsivun lisäksi toinen sivu, johon käyttäjällä on mahdollisuus lisätä Twitterin suosikkikäyttäjiään. Toiminnallisuus, jolla ohjelma siirtyy sivulta toiselle tehtiin ns. application bar -toiminnolla. Application bar ja sen näppäimet lisätään sovelluksen etusivulle XAML-koodia käyttäen:

```
<phone:PhoneApplicationPage.ApplicationBar>
<shell:ApplicationBar IsVisible="True" IsMenuEnabled="True"
BackgroundColor="#FF626262">
<shell:ApplicationBarIconButton Icon-
Uri="/Images/appbar.favs.addto.rest.png" Text="Favorites"
Click="btnGoFavorites_Click"/>
<shell:ApplicationBar.MenuItems>
<shell:ApplicationBarMenuItem Text="Favorites"/>
</shell:ApplicationBar.MenuItems>
</shell:ApplicationBar>
</phone:PhoneApplicationPage.ApplicationBar>
```

Koodin lisäämisen jälkeen etusivun alareunaan ilmestyy application bar -palkki, johon sovelluskehittäjä voi määrittää neljä eri näppäintoimintoa ja niille ikonit. Ikonit lisätään projektiin Solution Explorerista, ja niihin viitataan application bar -palkin XAML-koodissa IconUri kohdassa. Lisätyt ikonit eivät toimi sovelluksessa, ellei niiden asetuksista ole määritetty Build Action valinnaksi ”Content”. Build Actionin muuttaminen on esitelty kuvassa 26.



Kuva 26. Application bar ikonin Build Action valinnan muutos.

Application barin näppäimiin voidaan määrittää toimintoja vastaavasti kuin mihin tahansa muihin projektin näppäimiin. Ohjelmakoodissa määritetään mille sovelluksen sivulle näppäimen painaminen käyttäjän ohjaa:

```
private void btnGoFavorites_Click(object sender, EventArgs e)
{
NavigationService.Navigate(new Uri("/Favorites.xaml",
UriKind.Relative));
}
```

4.6.2 Suosikit

Uusi sivu, johon näppäimen painallus siirtää käyttäjän, voidaan tehdä lisäämällä Solution Explorerin kautta sovellukseen uusi Windows Phone Portrait Page. Sivun ulkoasun määrityksissä voidaan käyttää samaista XAML-koodia sekä samoja kontrolleja kuin pääsivulla. Uuden luodun sivun kontrollit sekä halutut tekstit pitää nimetä uudelleen päällekkäisyyksien estämiseksi.

Uusi sivu suunniteltiin sisältämään listauksen, johon käyttäjällä on mahdollisuus lisätä omia Twitter-palvelun suosikkikäyttäjiään. Listan nimeä napsauttamalla ohjelma hakee suoraan kyseisen henkilön twiitit. Tällä tavoin käyttäjän ei välttämättä tarvitse aina kirjoittaa hakusanaa, vaikkapa ollessaan hankalassa tilanteessa. Listan luominen onnistuu yksinkertaisesti koodilla:

```
public class Fav
{
          public string Name { get; set; }
}
```

Favorites sivun XAML-koodissa olevaan TextBlock -kontrolliin on myös vaihdettava oikea Binding Source, joka on tässä tapauksessa nimetty nimellä ”Name”:

```
<TextBlock Text="{Binding Name}" Margin="25,10,10,10" TextWrap-
ping="Wrap" FontSize="25" Width="350" FontWeight="Normal"
FontFamily="Segoe WP Semibold" />
```

Tällä tavoin lista saadaan näyttämään kaikki sinne lisätyt suosikit.

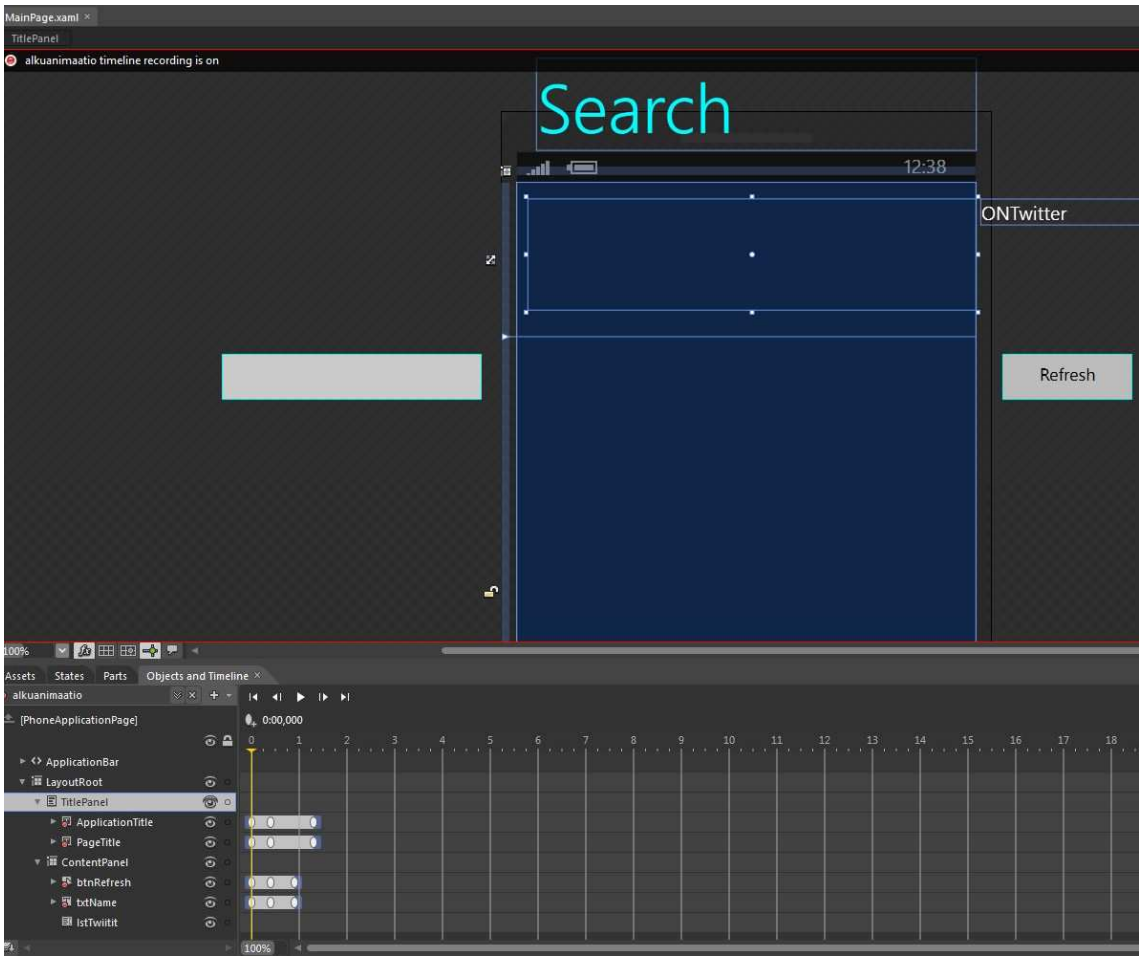
Valmis suosikit sivun käyttöliittymä on esitelty kuvassa 27.

Kuva 27. ONTwitter-sovelluksen suosikit sivu.

4.6.3 Animaatiot

ONTwitter-sovellus sisältää muutamia lisättyjä animointeja käyttökokemuksen parantamiseksi. Animaatioissa suunnittelun lähtökohtana oli se, että ne eivät saa lisätä ohjelman latausaikoja ja näin ollen hidastaa sovelluksen käyttöä. Sovelluksen animaatiot toteutettiin ajamaan itsensä aina kun sovellus ladataan käyttöön. Tällöin sovelluksen käynnistymisen yhteydessä käyttäjä näkee näppäinten ja muiden kontrollien siirtymisen sivulle animaatioina. Tällä saadaan

lisää interaktiivisuuden tunnetta sovellukseen sekä hienompi käyttökokemus. Animaatioiden luonti sovellukseen aikajanalla esitelty kuvassa 28.



Kuva 28. Animaatioiden luonti ONTwitter-sovellukseen.

Animaatio käynnistetään ohjelmakoodissa kutsumalla sitä MainPage -funktiossa. Kutsumisen jälkeen sovellus suorittaa animaation yhden kerran alusta loppuun asti.

5 JATKOKEHITYSMAHDOLLISUUDET

Windows Phone 7 -alustan mahdollisuudet tekevät sovelluksen käyttöliittymän jalostamisesta ja jatkokehittämisestä helppoa. Sovelluksen käyttöliittymä voidaan jälkikäteen siirtää suoraan käyttämään panoraama kontrollia. Tällöin sovelluksen virrankulutus lisääntyy hieman, mutta samalla ulkoasua voidaan kohentaa huomattavasti. Myös muita sivutyylejä voidaan lisätä käyttöliittymään tarpeen vaatiessa.

Autentikoinnin implementointi sovellukseen on yksi keskeisimmistä asioista jatkokehitystä ajatellen: Twitter-palvelun rajapinta mahdollistaakin kolmannen osapuolen sovelluksien kautta sisäänkirjautumisen. Tämä onnistuu OAuth Frameworkin avulla. OAuth antaa sovellukselle pääsyn niihin toimintoihin, joita esimerkiksi uusien twiittien kirjoittaminen vaatii.

Navigoinnin parantamiseksi sovelluksen sisällä jatkokehityksen kannalta tärkeäksi muodostuu myös Twitterin käyttäjänimien linkit. Tällä tarkoitetaan syötteestä saatujen käyttäjänimien muuttamista linkeiksi, joita painamalla pääsee suoraan kyseisen käyttäjän profiilisivulle. Näiden linkkien avulla sovelluksen käyttäjä pääsisi nopeasti tarkastelemaan eri Twitter palvelua käyttävien henkilöiden profiilisivuja. Nämä profiilisivut voisivat näkyä ONTwitter-sovelluksen sisällä kokonaan omina sivuinaan.

6 YHTEENVETO

Tämän opinnäytetyön tarkoituksena oli tarkastella mahdollisuuksia ja työkaluja, joita Windows Phone 7 -mobiilikäyttöjärjestelmä tuo sovelluskehittäjille. Näiden työkalujen avulla kehitettiin esimerkkisovellus, jolla pystyttiin osoittamaan yksinkertaisten työkalujen apu vaivattomassa sovelluskehityksessä.

Tehty esimerkkisovellus yhdistetään rajapinnan avulla Twitter-palveluun, josta saatiin ladattua puhelimeen palvelun sisältöä. Sovelluksen kehityksessä käytettiin Visual Studion lisäksi myös Expression Blend -työkalua. Työ esitti, kuinka helppoa työkalujen yhteiskäyttö on, ja kuinka yksinkertaisesti niiden avulla saa rikastettua sovelluksen käyttökokemusta. Kaiken tämän mahdollisti Windows Phone 7:n huolella suunniteltu alusta, joka huomioi niin loppukäyttäjän kuin kehittäjänkin.

Työn aikana tehty sovellus saatiin toimimaan kohtuullisen pienellä vaivalla juuri halutulla tavalla. Kuitenkin osa sovellukseen suunnitelluista ominaisuuksista jouduttiin jättämään pois niiden laajuuden takia. Sovellusta ei tulla julkaisemaan Microsoftin sovelluskaupassa. Tämä päätös johtuu Microsoftin tarjoaman kaupan 99 dollarin vuotuisesta rekisteröintimaksusta. Sovellusta ei myöskään rekisteröidä Twitter-palveluun, koska tämä vaatisi mm. erillisen internetsivuston toteuttamista sovellukselle.

LÄHTEET

[1] Järvinen Jani, Windows Phone: sovelluskehitys. Porvoo: Bookwell Oy, 2012.

[2] Windows Phone [www-dokumentti] Saatavilla: http://en.wikipedia.org/wiki/Windows_Phone (Luettu 06.2012).

[3] Windows Phone [www-dokumentti] Saatavilla: http://fi.wikipedia.org/wiki/Windows_Phone (Luettu 6.2012).

[4] Contact Us – Phone manufacturers [www-dokumentti] Saatavilla: http://www.microsoft.com/windowsphone/en-us/howto/contact-us.aspx (Luettu 8.2012).

[5] Mobile operating system – Market Share [www-dokumentti] Saatavilla: http://en.wikipedia.org/wiki/Mobile_operating_system (Luettu 8.2012).

[6] Matkapuhelimet [www-dokumentti] Saatavilla: http://www.gigantti.fi/catalog/puhelimet-ja-gps/fi_matkapuhelimet/matkapuhelimet (Luettu 7.2012).

[7] Application Platform Overview for Windows Phone [www-dokumentti] Saatavilla: http://msdn.microsoft.com/en-us/library/ff402531(v=VS.92).aspx (Luettu 7.2012).

[8] Windows Phone: Discover [www-dokumentti] Saatavilla: http://www.microsoft.com/windowsphone/en-us/features/default.aspx (Luettu 6.2012).

[9] Windows Phone: Pictures Hub [www-dokumentti] Saatavilla: http://www.microsoft.com/windowsphone/en-us/howto/wp7/pictures/pictures-hub.aspx (Luettu 8.2012).

[10] Multitasking for Windows Phone [www-dokumentti] Saatavilla: http://msdn.microsoft.com/en-us/library/windowsphone/develop/hh202866(v=vs.92).aspx (Luettu 8.2012).

[11] Additions in the Windows Phone SDK 7.1 [www-dokumentti] Saatavilla: http://msdn.microsoft.com/en-us/library/windowsphone/develop/ff637516(v=vs.92).aspx (Luettu 8.2012).

[12] About Silverlight [www-dokumentti] Saatavilla: http://www.microsoft.com/silverlight/what-is-silverlight/ (Luettu 7.2012).

[13] Features Differences Between Silverlight and Silverlight for Windows Phone [www-dokumentti]
Saatavilla: http://msdn.microsoft.com/en-us/library/ff426931%28v=VS.95%29.aspx (Luettu 7.2012).

[14] XNA-peliohjelmointiympäristö [pdf-dokumentti] Saatavilla:
http://www.cs.helsinki.fi/u/vihavain/k08/sem/korjatut/maaranen_timo_XNA_peliohjelmointiymparisto.pdf (Luettu 7.2012).

[15] Microsoft XNA [www-dokumentti] Saatavilla: http://en.wikipedia.org/wiki/Microsoft_XNA (Luettu 7.2012).

[16] Windows Phone Reaches The 100,000 Application Milestone [www-dokumentti] Saatavilla: http://www.forbes.com/sites/ewanspence/2012/06/05/windows-phone-reaches-the-100000-application-milestone/ (Luettu 8.2012).

[17] Windows Phone Dev Center – Join [www-dokumentti] Saatavilla: https://dev.windowsphone.com/en-us/join (Luettu 8.2012).

[18] Submitting a Windows Phone 7 Application to the Market. [www-dokumentti] Saatavilla: http://geekswithblogs.net/mbcrump/archive/2010/11/09/submitting-a-windows-phone-7-application-to-the-market.aspx (Luettu 8.2012).

[19] About – About Twitter [www-dokumentti] Saatavilla: https://twitter.com/about (Luettu 9.2012).

[20] Ohjelmointirajapinta [www-dokumentti] Saatavilla: http://suomisanakirja.fi/ohjelmointirajapinta (Luettu 9.2012).

[21] The Streaming APIs – Differences between Streaming and REST [www-dokumentti] Saatavilla: https://dev.twitter.com/docs/streaming-apis (Luettu 9.2012).